自 然 言 語 処 理 152－11
（２００２．１１．１２）

SD 式意味モデルにおける概念体系を利用した単文の言い換えと評価方法の提案

峯脇 さやか　　新見 道治　　河口 英二

「言い換え」とは，同じ意味内容を表す複数の表現を結びつける変換である．本研究では，著者らが提案している SD 式意味モデルを利用した単文の言い換え手法と生成した言い換え文の評価方法を提案する．SD 式意味モデルとは，意味表現形式 SD 式による意味処理を行なうモデルであり，概念間の意味的な近さを定量的に扱えるという特徴をもつ．本研究では，ある SD 式を別の構造の SD 式に変換することで，意味レベルでの言い換えを試みる．言い換え生成では，概念体系における概念の上位－下位関係を利用する．評価では，言い換えられた概念が，抽象化／同義／具体化のいずれになるかを定量的に判定する．本稿では，言い換え生成・評価方法および，実験例を示す．

A Simple Sentence Paraphrase Using Concept Hierarchy in SD-Form Semantics Model

Sayaka MINEWAKI, Michiharu NIIMI and Eiji KAWAGUCHI

This paper proposes a method of a simple sentence paraphrase and a framework for the evaluation of paraphrased sentences. The method uses the frameworks of SD-Form Semantics Model that has been developed by the authors. The paraphrase is defined as SD-Form~to~SD-Form transformations, which is based on the concept hierarchy in SD-Form Semantics Model. In the evaluation, paraphrased concepts are categorized into abstract concept, coordinate or embodiment concept. In this paper, we showed a concrete way to paraphrase and some examples.

1. はじめに

「言い換え」とは，同じ意味内容を表す複数の表現を結びつける変換で，人間の言語処理能力の一部である[1]．我々人間は，話し言葉・書き言葉にかかわらず，ある概念を同じ意味（同義）のまま別の表現に変えるだけでなく，抽象化したり，具体化したりできる．

言い換えは，自然言語処理の様々な分野で，前処理や目的実現のための技術の一つとして用いられてきた[2]．機械翻訳では，機械処理に適した表現にあらかじめ書き換えておいたり，適格でない表現を修正したりと翻訳の前後で言い換えを利用できる[3]．自動要約は，重要な情報を残し，かつ文字数を減らすという目的をもった言い換えである．そして，言い換えを利用した情報検索が提案されている[4]．さらに言い換えは，テキストベースの情報秘匿技術にも応用できる[5],[6]．

このような背景から，言い換えの重要性に気付き，近年，独立した技術として研究されるようになってきた．しかし，研究事例はまだ少なく，すでに可能となった技術ではない[1],[2],[3]．言い換えは，これから実現すべく努力すべき研究対象である[3]．文献[1]では，言い換えの種類を「構文的言い換え」,「意味的言い換え」,「プラグマティックな言い換え」の 3 種類に分類する考え方を示している．このうち最も実現可能なものが構文的言い換えであり，言い換えに関する研究のほとんどが構文的言い換えを行なっている[1],[2]．構文的言い換えは，世界知識や文脈に関係なく，言語学的知識だけで実現できるため，容易に見える．しかし，人間は言語知識に頼った言い換えだけでなく，意味情報に基づいた言い換えも行なう．表層構造のみの変換を行なう構文的言い換えだけでは，人間と同等の言い換えはできない．これからは，深層構造に着目した言い換えの研究が必要である．

そこで，著者らの研究グループが自然言語の意味表現形式の一つとして提案している SD 式（Semantic-structure Description Form）[7]を

九州工業大学工学部
Faculty of Engineering, Kyusyu Institute of Technology

利用した言い換えモデルを提案する．SD 式は，自然言語における個々の概念，陳述表現，感情表現，あるいはシステムに与える知識データなどを記述するための一種の中間言語である．自然言語概念を SD 式として捉え，その記述データを基にして意味処理を行なおうとするモデルが SD 式意味モデルである．SD 式意味モデルでは，概念間の意味的な近さを定量的に扱うことができるという特徴をもつ．

本研究では，ある SD 式を別の構造の SD 式に変換することで，意味レベルでの言い換えを試みる．本稿で提案する言い換え生成手法では，意味ネットワーク（概念体系）における概念の上位－下位関係を利用する．入力に最も近い概念を意味ネットワーク中から見出し，見出した概念およびその上位概念･下位概念を入力の言い換えとする．

ある SD 式を別の構造の SD 式に変換することは，ある概念をその上位概念または下位概念に変換することである．上位概念に変換したことで，もとの概念は抽象化され，また，下位概念に変換したことで，もとの概念を具体化されたことになる．言い換えられた概念が言い換えた概念を抽象化したものなのか，あるいは具体化したものが明確になれば，言い換えを用いたアプリケーションにとって有効である．そこで，生成された言い換えが，「抽象化した概念」，「同義の概念」，「具体化した概念」のいずれになるかを SD 式意味モデルで定義された枠組を用いて定量的に判定し，これを評価とする．

以下，**2.**では，SD 式意味モデルの概要について述べる．**3.**では，概念体系を利用した単文の言い換え手法と評価方法について述べ，**4.**で処理手順を例示する．最後に **5.**でまとめと今後の課題について述べる．

2. SD 式意味モデルの概要[7]

SD 式意味モデル（Semantic-structure Description Form Semantics Model）は，自然言語の意味を定量的に分析するための枠組である．このモデルに従った意味記述を SD 式と呼ぶ．SD 式は，自然言語における個々の概念，陳述表現，感情表現，システムに与える知識データなどを記述したものである．SD 式意味モデルの特徴は，与えられた 2 つの概念の意味的な差異を定量的に扱えることである．

2.1 SD 式の記述例

SD 式は，文脈自由言語であり，①概念ラベル，②規定子，③結合子，④修飾子，⑤機能項目記号，⑥区切り記号の 6 種類から構成される記号列である．これらを「SD 式記号」と呼ぶ．概念ラベルは既成の単語（日本語や英語）を使用しており，変数ラベル，単純ラベルなどに分類している．機能項目記号は，意味上の機能を表すもので，主語項目 *s*(D)を，述語項目を *v*(D)，…としている．このときの D を機能項目の内容と呼ぶ．

表 2.1　SD 式の記述例

(a) 陳述 SD 式

SD 式	自然言語
[*s*(自分),*v*(テニス/時/毎日)]	私は，毎日テニスをする．
[*s*(トム),*v*(である),*c*(アメリカ人)]	トムは，アメリカ人である．
[*s*(美紀),*v*(貰う/過去),*o*(花)]	美紀は，花を貰った．

(b) 感情 SD 式

SD 式	自然言語
[*a*(ジョン)]	ジョン．（呼びかけ）
[*r*(否定)]	いいえ．（応答）
[*e*(驚き)]	わぁ！（感嘆）

表 2.2　知識データの記述例

SD 式	自然言語
(九州)*incl*(沖縄)	九州は沖縄を含む．
(りんご)*ptof*(果物)	りんごは果物の一種である．

表 2.3　結合子の種類と用法

種類	用法
defi	定義
equa	等価
incl	包含

種類	用法
kdof	種類
para	並列関係
ptof	部分

SD 式は，英語の 5 文型を模した陳述 SD 式と，感情的な発声や発話を記述する感情 SD 式に大別できる．陳述 SD 式および感情 SD 式の記述例を表 2.1 に示す．

知識データの記述例を表 2.2 に示す．表 2.2 にある“ *incl* ”,“ *ptof* ”は SD 式記号の結合子である．結合子の種類と用法を表 2.3 に示す．

2.2 SD 式の意味的情報量

意味を定量的に処理するために，各 SD 式記号に意味素量と呼ばれる値が与えられている．ある SD 式に対して，その SD 式を構成する SD 式記号の意味素量を全て足し合わせた値を，その SD 式の「意味量」として定義している．任意の SD 式を D とするとき，その意味量を

$$si(\mathrm{D}) = n \quad [semit]$$

と表し，その単位を *semit* としている．以下に意味素量の例を示す．

(1)変数ラベル　“ X, Y, Z, ... ”　：1[*semit*]
(2)単純ラベル　“ 車, 買う,… ”　：10[*semit*]
(3)修飾子　“ / ”　：1[*semit*]
(4)規定子　“ *nega*, *only*,… ”　：2[*semit*]
(5)結合子　“ *para*, *equa*,… ”　：1[*semit*]
(6)機能項目記号　“ *s*, *v*, *o*,… ”　：1[*semit*]
(7)区切り記号　“ [ ] ”　：1[*semit*]
(8)区切り記号　“ ( ) ”, “ , ”　：0[*semit*]

SD 式の意味量は，その SD 式を構成する各記号の意味素量の総和である．以下に，意味量の例を示す．

＜例 2.1＞

si(帽子/赤い) = 21
si([*s*(相手),*v*(読む/過去),*o*(本/当該)]) = 56
si([*e*(感嘆/賞賛)]) = 23

2.3　2 概念間の詳述関係

2.3.1　詳述関係の定義

2 つの概念 D_1，D_2 に関して，D_1 の意味をより具体化したものが D_2 であり，かつ，D_2 の意味をより抽象化したものが D_1 であるとき，「D_1 と D_2 には詳述関係がある」という．このとき，D_1 を D_2 の先祖，D_2 を D_1 の子孫と呼ぶ．D_1 から D_2 への詳述関係を

$$elab(D_1,D_2) = n$$

と表す．ここで，n（$0 \le n < \infty$）は詳述量といい，詳述の程度を表すものである．

詳述関係には，SD 式の構造による「構文的詳述関係」と，システムが利用できる知識データにも基づく「知識に基づく詳述関係」があり，それぞれ

$$elab_{synt}(D_1,D_2) = n$$
$$elab_{know}(D_1,D_2) = n$$

と表す．一般的には，

$$elab(D_1,D_2) = min\{elab_{synt}(D_1,D_2), elab_{know}(D_1,D_2)\}$$

と定義している．

2.3.2　構文的詳述関係

2 つの概念 D_1，D_2 について，D_1 から D_2 への詳述関係のうち，D_1 と D_2 の構文により成り立つ場合を，構文的詳述関係という．このときの詳述量は次のように定義している．

$$elab_{synt}(D_1,D_2) = si(D_2) - si(D_1)$$

図 2.1 に 2 つの陳述 SD 式における詳述関係を示す．構文的詳述関係における詳述量の計算例を以下に示す．

＜例 2.2＞

(1) $elab_{synt}$(時計,時計/(古い)*para*(大きい))
= *si*(時計/(古い)*para*(大きい)) – *si*(時計)
= 22

(2) $elab_{synt}$([*s*(人間),*v*(食べる)],
[*s*(人間),*v*(食べる),*o*(野菜)])
= 11

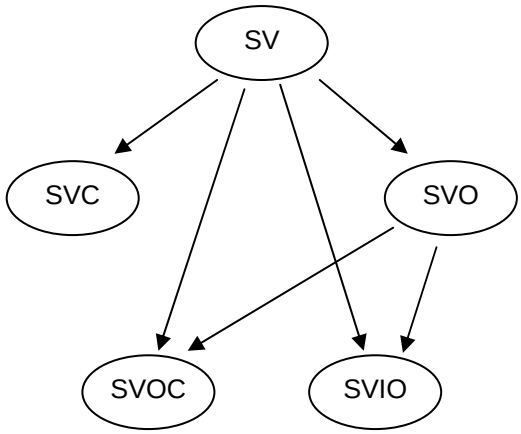

図 2.1　陳述形式の D_1 と D_2 において，
D_1 が D_2 の一部となる場合の関係：
矢印の始点が先祖で終点が子孫の関係

2.3.3　知識に基づく詳述関係

与えられた 2 つの概念 D_1，D_2 自身には，構文的な詳述関係がない場合でも，D_1，D_2 を特別な関係で結びつける知識データがあれば詳述関係が生じてくる．

知識に基づく詳述関係には，(1)個別の知識に基づく詳述関係と，(2)一般の知識に基づく詳述関係がある．

(1)個別の知識に基づく詳述関係

個別の知識に基づく詳述関係は，システムに登録された次のような知識データによって定まる．

$(D_1)equa(D_2)$　→　$elab_{know}(D_1,D_2) = 0$
$(D_1)defi(D_2)$　→　$elab_{know}(D_1,D_2) = 0$
$(D_1)incl(D_2)$　→　$elab_{know}(D_1,D_2) = 3$
$(D_1)ptof(D_2)$　→　$elab_{know}(D_1,D_2) = 3$
$(D_1)kdof(D_2)$　→　$elab_{know}(D_1,D_2) = 3$

個別の知識に基づく詳述関係における詳述量の計算例を以下に示す．

＜例 2.3＞

システムに次の知識データを与えた場合，
(首都/日本)*equa*(東京)
次のような詳述関係が成り立つ．
$elab_{know}$(首都/日本,東京) = 0

(2)一般の知識に基づく詳述関係

一般の知識に基づく詳述関係は，システムに組み込まれた“ 一般に成り立つと考えられる ”次のような規則によって定まる．

$elab_{know}(D,nega(nega(D))) = 2$

$elab_{know}$(D/X,D/SOME) = 1

2.4 意味差の尺度

SD 式意味モデルでは，概念間の「意味差の尺度」を次のように定義している．2 つの概念 D_1, D_2 に共通する全ての先祖 $D_{01}, D_{02}, \ldots$の中で D_1, D_2 に最も近い先祖を「D_1，D_2 の最近共通先祖」と呼び，D_0 で表す．この関係を

$$ncoa(D_1,D_0,D_2)$$
$$= elab(D_0,D_1) + elab(D_0,D_2)$$
$$= min_i\{elab(D_{0i},D_1) + elab(D_{0i},D_2)\}$$
$$= n_0$$

と表す．このときの n_0 を「$\mathbf{D_1}$ と $\mathbf{D_2}$ の意味差」といい，

$$diff(D_1,D_2) = n_0$$

と表す．すなわち，意味差は与えられた $\mathbf{D_1}$，$\mathbf{D_2}$ の最近共通先祖を探索することにより求められる．

2.5 概念の体系化

概念の体系化とは，詳述関係に基づいて SD 式を結びつけることである．概念の体系化を行なうことで，概念の上位－下位関係の階層構造が得られる．

概念体系とは，概念集合と詳述量の集合からなる意味ネットワークである．概念集合のそれぞれの概念を節，詳述量を枝とすることで，概念の体系化を表すことができる．また，概念体系において，最上位概念を変数ラベル X としている．

概念の体系化の例を以下に示す．

＜例 2.4＞

表 2.4 に示す知識データ F_1～F_3 がシステムに登録されているとする．このときの概念体系を図 2.2 に示す．ここで，図 2.2 のΩは概念集合を表す．つまり，

$$\Omega = \{X, F_1, F_2, F_3\}$$

である．

2.6 認識

SD 式意味モデルでは，**2.5** で述べた概念体系を利用することにより，認識・理解・解釈の知的処理が可能となる．本稿では，認識動作のみについて述べる．

SD 式意味モデルにおいて形式化した認識とは，与えられた入力をシステムの既知概念の一つに属するものと結論することである．

外部入力として，ある概念 D_{in} が与えられたとする．D_{in} を認識するとは，システム内の概念体系において，D_{in} に最も近い先祖 D_{rec} を見出すことである．D_{rec} は次式を与える D である．

$$min_D\{elab(D,D_{in}) \mid D \in \Omega\}$$

ただし，Ωは概念集合である．このとき D_{in} と D_{rec} の関係を次のように表す．

$$recog(D_{in},D_{rec})$$

表 2.4　例 2.4 の知識データ

	SD 式	自然言語
F_1	刺身	刺身
F_2	魚料理	魚料理
F_3	(刺身)*kdof*(魚料理)	刺身は魚料理の一種である．

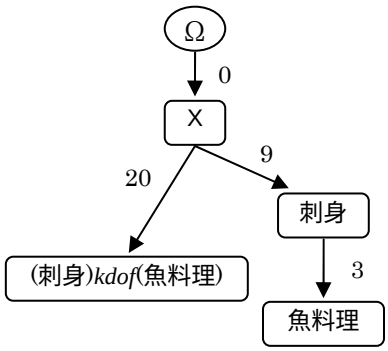

図 2.2　概念体系図

3. 概念体系を利用した単文の言い換え

本研究では，ある SD 式を別の構造の SD 式に変換することで，意味レベルでの言い換えを試みる．本稿で提案する言い換え生成手法では，概念体系における概念の上位－下位関係を利用する．あらかじめシステムに存在する概念体系において，まず言い換える SD 式を認識し，見出した概念およびその上位概念・下位概念を生成する言い換えとする．評価では，生成された言い換えが，「抽象化した概念」,「同義の概念」,「具体化した概念」のいずれになるかを意味量，意味差，詳述量を順次用いて定量的に判定する．

以下，言い換え生成手法と評価手順についてそれぞれ **3.1**，**3.2** で詳しく述べる．

3.1 言い換え生成手法

あらかじめシステムに存在する概念体系を利用した単文の言い換え生成手順を以下に示す．

Step-1　認識

言い換えの対象となる SD 式を D_{in} とする．D_{in} を認識し，確定した概念を D_{rec} とする（**2.6** 参

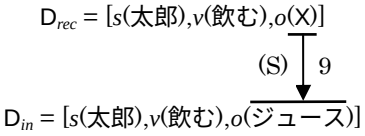

図 3.1　D_{rec} と D_{in} の詳述関係：
(S)は構文的詳述関係を示す．

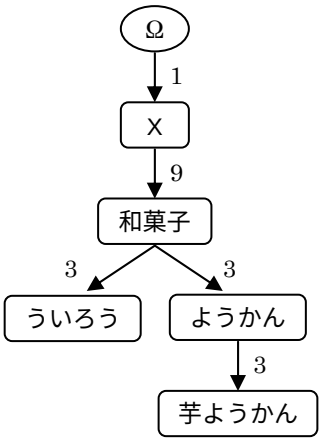

図 3.2　F_1～F_4 の概念体系図

照)．ただし，D_{rec}≠X である．

Step-2　変数ラベルの具現化

D_{rec} 中に変数ラベルがある場合，D_{rec} 中の変数ラベルを D_{in} 中の詳述関係を持つ SD 式に一致させる．この操作を，変数ラベルの具現化と呼ぶ．

Step-3　言い換え生成

D_{rec} の全ての先祖を HC_P（p=1,2,…）とし，また，全ての子孫を LC_q（q=1,2,…）とする．ただし，HC_p≠X とする．D_{rec}，HC_P，LC_q を順次基本概念として，言い換えを生成する．まず，基本概念を言い換えとする．次に，基本概念の機能項目の内容を D_i（i=1,2,…）とし，概念体系中にある D_i の先祖や子孫を D_{ij}（j=1,2,…）とする．ただし，D_{ij}≠X とする．D_i と D_{ij} を置換することで言い換えを生成する．生成された言い換えが，D_{in} と同じ SD 式でないならば，これを P_k（k=1,2,…）とする．

変数ラベルの具現化の例を例 3.1 に，**Step-3** における D_i と D_{ij} を置換の例を例 3.2 示す．

＜例 3.1＞

次の D_{rec} と D_{in} の詳述関係を図 3.1 示す．

D_{rec} = [s(太郎),v(飲む),o(X)]

D_{in} = [s(太郎),v(飲む),o(ジュース)]

このとき，D_{rec} 中の変数ラベル X を具現化すると，

X = ジュース

表 3.1　例 3.2 の知識データ

	SD 式	自然言語
F_1	和菓子	和菓子
F_2	ういろう	ういろう
F_3	ようかん	ようかん
F_4	芋ようかん	芋ようかん
F_5	(和菓子)*incl*(X)	和菓子は X を含む.
F_6	(和菓子)*incl*(ういろう)	和菓子はういろうを含む.
F_7	(和菓子)*incl*(ようかん)	和菓子はようかんを含む.
F_8	(ようかん) *incl*(芋ようかん)	ようかんは芋ようかんを含む.
F_9	[s(太郎),v(好む), o(ようかん)]	太郎はようかんを好む.

となる．よって，D_{rec} を次のようにする．

D_{rec} = [s(太郎),v(飲む),o(ジュース)]

＜例 3.2＞

表 3.1 に示す知識データ F_1～F_9 がシステムに登録されているとする．このときの F_1～F_4 のみの概念体系を図 3.2 に示す．言い換えの基本概念を F_9 とする．F_9 の機能項目の内容は以下のとおりである．

D_1 = 太郎　D_2 = 好む　D_3 = ようかん

上記の知識データに D_1，D_2 の先祖および子孫が存在しないことは明らかである．D_3 の先祖および子孫は次のようになる．

D_{31} = 和菓子　D_{32} =芋ようかん

D_3 を D_{31}，D_{32} と置換すると，次のような言い換えが得られる．

P_1 = [s(太郎),v(好む),o(和菓子)]
：太郎は和菓子を好む.

P_2 = [s(太郎),v(好む),o(芋ようかん)]
：太郎は芋ようかんを好む.

3.2　評価手順

提案手法における評価では，言い換え P_k が D_{in} を「抽象化した言い換え」，「同義の言い換え」，「具体化した言い換え」のいずれになるかを，意味量，意味差，詳述量を順次用いて判定する．評価手順を以下に示す．

Step-1　P_k と D_{in} の意味量の大小に着目する．

(a) $si(P_k) < si(D_{in})$のとき，抽象化.
(b) $si(P_k) > si(D_{in})$のとき，具体化.
(c) $si(P_k) = si(D_{in})$のとき，(2)へ.

Step-2　P_k と D_{in} の意味差に着目する．

(a) $diff(P_k,D_{in}) = 0$ のとき，同義.
(b) $diff(P_k,D_{in}) \neq 0$ のとき，(3)へ.

Step-3　P_k, D_{in} と最近共通先祖 D_0 とのそれぞれ

表 4.1　知識データ

	SD 式	自然言語
F_1	場所	場所
F_2	ありか	ありか
F_3	(場所)*equa*(ありか)	場所とありかは等しい.
F_4	場所/X1	X1 の場所
F_5	ありか/X1	X1 のありか
F_6	(場所/X1)*equa*(ありか/X1)	X1 の場所と X1 のありかは等しい.
F_7	場所/[*s*(X1),*v*(存在)]	X1 が存在する場所.
F_8	(場所/X1)*incl*(場所/[*s*(X1),*v*(存在)])	X1 の場所は X1 が存在する場所を含む.
F_9	[*s*(自分),*v*(質問),*o*(相手),*c*([*s*(X1),*v*(存在/場所/何処)])]	X1 は何処にありますか？
F_{10}	[*s*(自分),*v*(質問),*o*(相手($1)),*c*([*s*($1),*v*(知る),*o*(場所/X1)])]	X1 の場所を知りませんか？
F_{11}	(F_9)*incl*(F_{10})	F_9 は F_{10} を含む.
F_{12}	[*s*(自分($1)),*v*(依頼),*o*(相手($2)),*c*([*s*($2),*v*(教える),*i*($1),*o*(場所/X1)])]	X1 の場所を私に教えてくれませんか？
F_{13}	(F_{10})*incl*(F_{12})	F_{10} は F_{12} を含む.

の詳述量の大小に着目する.

(a) $elab(D_0,P_k) < elab(D_0,D_{in})$のとき，抽象化.

(b) $elab(D_0,P_k) > elab(D_0,D_{in})$のとき，具体化.

(c) $elab(D_0,D_{in}) = elab(D_0,P_k)$のとき，判定不能.

評価で用いる各枠組について, それぞれ次のような性質を利用している.

(1) 意味量

意味量は SD 式の構造に依存する．SD 式が簡単な構造である場合，意味量は小さく，複雑である場合，意味量は大きい．SD 式が簡単な構造であるときは，より抽象的な概念を表し，複雑なときは，より具体的な概念を表している.

(2) 意味差

意味差は，2 つの概念間の意味的な差異である. 意味的な差異が 0 である場合, その 2 つの概念は同義であると見なすことができる.

(3) 詳述量，最近共通先祖

(1)では，意味量という大きさに着目し，意味量が等しい場合，(2)で 2 つの概念間の距離に着目した．(3)では，最近共通先祖という基準を設けて，D_0 と P_k，D_0 と D_{in} の詳述量をそれぞれ計算し，どちらが最近共通先祖に近いか遠いかで，抽象化／具体化の判定をする.

評価手順の例を示す.

＜例 3.3＞

例 3.2 の F_9 を D_{in} として，得られた言い換え P_1 の評価を行なう.

まず，2 つの意味量の大小を比較する．P_1 と D_{in} の意味量は次のようになる.

$si(P_1) = 34$

$si(D_{in}) = 34$

2 つの意味量は等しいので, 次に P_1 と D_{in} の意味差を計算する．P_1 と D_{in} の最近共通先祖 D_0 は次のようになる.

D_0 = [*s*(太郎),*v*(好む),*o*(和菓子)]

よって意味差は次のようになる.

$$diff(P_1,D_{in}) = elab(D_0,P_1) + elab(D_0,D_{in})$$
$$= 0 + 3$$
$$= 3 \neq 0$$

意味差が 0 でないので, 最近共通先祖からの詳述量の大小を比較する. それぞれの詳述量は次のようになる.

$elab(D_0,P_1) = 0$

$elab(D_0,D_{in}) = 3$

$elab(D_0,P_1) < elab(D_0,D_{in})$なので，$P_1$ の方が D_{in} よりも D_0 に近い．よって，P_1 は抽象化した言い換えと判定する.

4.　言い換え生成過程の例示

3.で示した手順に従って，言い換え生成過程および評価を例示する.

表 4.1 に示す知識データ F_1～F_{13} がシステムに登録されているとする. このときの概念体系図を図 4.1 に示す.

概念体系化の過程で次の新たな概念が導出されている.

(X)*incl*(Y)　　：X は Y を含む.

言い換える SD 式 D_{in} は次のものとする.

D_{in} = [*s*(自分),*v*(質問),*o*(相手($1)),
c([*s*($1),*v*(知る),*o*(ありか/gcc ソース)])]

：gcc ソースのありかを知りませんか？

言い換え生成過程を以下に示す.

Step-1

D_{in} を認識する．図 4.1 に示す概念体系において，D_{in} に最も近い先祖は F_{10} である.

$recog(D_{in},F_{10})$　　($D_{rec} = F_{10}$)

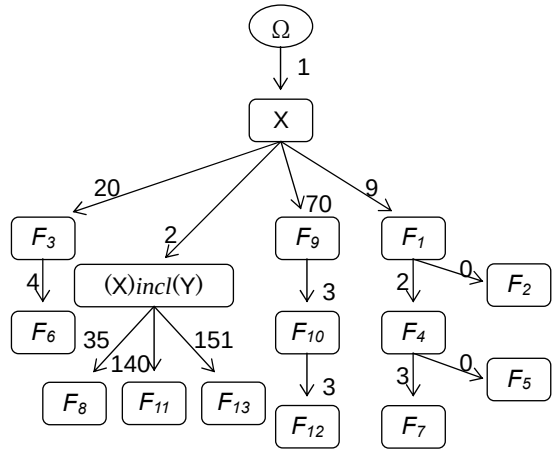

図 4.1　概念体系図

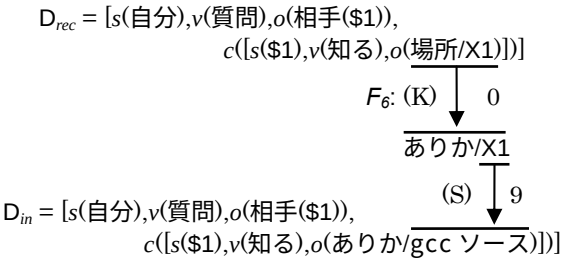

図 4.2　D_{rec} と D_{in} の詳述関係：
(S)は構文的詳述関係,
(K)は知識に基づく詳述関係を表す.

Step-2

D_{rec} の変数ラベル X1 を具現化すると,

X1 = gcc ソース

となる．図 4.2 に D_{rec} と D_{in} の詳述関係を示す．X1 を具現化した D_{rec} は次のようになる．

D_{rec} = [*s*(自分),*v*(質問),*o*(相手($1)),
c([*s*($1),*v*(知る),*o*(場所/gcc ソース)])]
：gcc ソースの場所を知りませんか？

Step-3

D_{rec} の先祖 HC_p および子孫 LC_q は次のとおりである．

$HC_1 = F_9$　$LC_1 = F_{12}$

まず，D_{rec} を基本概念として言い換えを生成する．$D_{rec} \neq D_{in}$ なので，次の言い換えを得る．

P_1 = [*s*(自分),*v*(質問),*o*(相手($1)),
c([*s*($1),*v*(知る),*o*(場所/gcc ソース)])]
：gcc ソースの場所を知りませんか？

D_{rec} の機能項目の内容 D_i は次のとおりである．

D_1 = 自分　D_2 = 質問　D_3 = 相手
D_4 = 知る　D_5 = 場所/X1

D_1〜D_5 において，図 4.1 に示す概念体系中に先祖および子孫を持つものは，D_5 のみである．D_5 の先祖および子孫 D_{5j} を次に示す．

$D_{51} = F_1$ = 場所　$D_{52} = F_2$ = ありか
$D_{53} = F_5$ = ありか/X1
$D_{54} = F_7$ = 場所/[*s*(X1),*v*(存在)]

D_5 を D_{51}, D_{52}, D_{53}, D_{54} にそれぞれ置換すると，次のような言い換えを得る．

P_2 = [*s*(自分),*v*(質問),*o*(相手($1)),
c([*s*($1),*v*(知る),*o*(場所)])]
：場所を知りませんか？

P_3 = [*s*(自分),*v*(質問),*o*(相手($1)),
c([*s*($1),*v*(知る),*o*(場所)])]
：ありかを知りませんか？

[*s*(自分),*v*(質問),*o*(相手($1)),
c([*s*($1),*v*(知る),*o*(ありか/gcc ソース)])]
：gcc ソースのありかを知りませんか？

P_4 = [*s*(自分),*v*(質問),*o*(相手($1)),
c([*s*($1),*v*(知る),
o(場所/[*s*(gcc ソース),*v*(存在)])])]
：gcc ソースがある場所を知りませんか？

ここで，生成された SD 式のうち 3 番目の SD 式は，D_{in} と同じなので P_k としない．

次に，HC_1 を基本概念として言い換えを生成する．X1 = gcc ソースより，次の言い換えを得る．

P_5 = [*s*(自分),*v*(質問),*o*(相手),
c([*s*(gcc ソース),*v*(存在/場所/何処)])]
：gcc ソースはどこにあるのですか？

HC_1 の機能項目の内容は，全て先祖および子孫を持たないので，HC_1 からの言い換え生成は終了する．

そして，LC_1 を基本概念として言い換えを生成する．X1 = gcc ソースより，次の言い換えを得る．

P_6 = [*s*(自分($1)),*v*(依頼),*o*(相手($2)),
c([*s*($2),*v*(教える),*i*($1),*o*(場所/gcc ソース)])]
：gcc ソースの場所を教えてくれませんか．

LC_1 の機能項目の内容で先祖および子孫を持つものは，場所/X1 のみである．これは，D_{rec} のときと同じである．LC_1 から次の言い換えを得る．

P_7 = [*s*(自分(\$1)),*v*(依頼),*o*(相手(\$2)),
c([*s*(\$2),*v*(教える),*i*(\$1),*o*(場所)])]
：場所を教えてくれませんか.

P_8 = [*s*(自分(\$1)),*v*(依頼),*o*(相手(\$2)),
c([*s*(\$2),*v*(教える),*i*(\$1),*o*(ありか)])]
：ありかを教えてくれませんか.

P_9 = [*s*(自分(\$1)),*v*(依頼),*o*(相手(\$2)),
c([*s*(\$2),*v*(教える),*i*(\$1),*o*(ありか/gcc ソース)])]
：gcc ソースのありかを教えてくれませんか.

P_{10} = [*s*(自分(\$1)),*v*(依頼),*o*(相手(\$2)),
c([*s*(\$2),*v*(教える),*i*(\$1),
o(場所/[*s*(gcc ソース),*v*(存在)])])]
：gcc ソースがある場所を教えてくれませんか.

以上より，10 個の言い換え P_1～P_{10} が生成できた．図 4.3 に D_{in} と P_1～P_{10} をそれぞれ自然言語文に変換したものを示す.

表 4.2 に P_1～P_{10} の評価を示す. D_{in} の意味量は 80[*semit*]である．P_1，P_5，P_7，P_8 の最近共通先祖は次のとおりである.

ncoa(P_1,P_1,D_{in})　（$D_0 = P_1$）
ncoa(P_5,P_5,D_{in})　（$D_0 = P_5$）
ncoa(P_7,X,D_{in})　（D_0 = X）
ncoa(P_8,X,D_{in})　（D_0 = X）

5. おわりに

本稿では，SD 式意味モデルにおける概念体系を利用した言い換えモデルと評価方法について述べ, 言い換え生成過程および評価について例示した. 生成された概念を自然言語文に変換したものは，全て自然な言い換えであるといえる.

今後の課題は, プロトタイプシステムを作成し, 本手法の妥当性を検証することである.

D_{in} ：gcc ソースのありかを知りませんか？
P_1 ：gcc ソースの場所を知りませんか？
P_2 ：場所を知りませんか？
P_3 ：ありかを知りませんか？
P_4 ：gcc ソースがある場所を知りませんか？
P_5 ：gcc ソースはどこにあるのですか？
P_6 ：gcc ソースの場所を教えてくれませんか.
P_7 ：場所を教えてくれませんか.
P_8 ：ありかを教えてくれませんか.
P_9 ：gcc ソースのありかを教えてくれませんか.
P_{10} ：gcc ソースがある場所を教えてくれませんか.

図 4.3　D_{in}，P_1～P_{10} の自然言語文

表 4.2　評価（*si*(D_{in}) = 80[*semit*]）

	意味量	意味差	詳述量	評価
P_1	80	0	－	同義
P_2	69	－	－	抽象化
P_3	69	－	－	抽象化
P_4	93	－	－	具体化
P_5	80	≠ 0	*elab*(D_0,P_5) < *elab*(D_0,D_{in})	抽象化
P_6	91	－	－	具体化
P_7	80	≠ 0	*elab*(D_0,P_7) > *elab*(D_0,D_{in})	具体化
P_8	80	≠ 0	*elab*(D_0,P_8) > *elab*(D_0,D_{in})	具体化
P_9	91	－	－	具体化
P_{10}	104	－	－	具体化

参考文献

[1] 佐藤　理史：“ 論文表題を言い換える ”, 情報処理学会論文誌，Vol.40，No.7，pp.2937-2945，1999
[2] 近藤　恵子，佐藤　理史，奥村　学：“ 格変換による単文の言い換え ”, 情報処理学会論文誌, Vol.42, No.30，pp.465-477，2001
[3] 乾　健太郎：“ 言語表現を言い換える ”, 言語処理学会第 8 回年次大会チュートリアル，2002
[4] P. Wallis: “Information Retrieval based on Paraphrase”, PACLING, 1993
[5] 中川　裕志，三瓶　光司，松本　勉，柏木　健志，川口　修司，牧野　京子，村瀬　一郎：“ 意味保存型の情報ハイディング―日本語文書への適用 ”, 情報処理学会論文誌, Vol.42，No.9，pp.2339-2350，2001
[6] 峯脇　さやか, 伊藤　友和, 新見　道治, 野田　秀樹，河口　英二：“ SD 式を利用した言語ステガノグラフィ ”，情報処理学会研究報告，Vol.2002, No.68, 2002-CSEC-18，pp.137-144，2002
[7] M. Wakiyama, H. Noda, K. Nozaki, and E. Kawaguchi: "Computation Algorithm of Semantic Difference Measure in the SD-Form Semantics Model" Trans. IPSJ, Vol.40, No.3, pp.1065-1079, 1999