SD式意味モデルをWeb上で実験するための試み

吉原 将太・脇山 正博・河口 英二

Try to Experiment on the SD-Form Semantics Model on Web

Shota Yoshihara, Masahiro Wakiyama and Eiji Kawaguchi

Abstract

The SD-Form Semantics Model, developed by the authors, is a framework to deal with the meaning of natural language in a quantitative way. It is associated with a formal language named SD-Form that describes the semantic structure of each language expression.

We implemented an experimental system (SDENV-3) by Prolog for feasibility study. However, it is difficult to combine Prolog programs as other application systems. Then, We decided to program the SDENV with Perl to experiment on Web. The system is called SDENV-4.

The objective of the present paper is to show the SDENV-4. The system operation is shown in brief by illustrating an example problem.

1．はじめに

計算機とは，与えられた規則に従って記号の操作を行うものである。しかし，自然言語(natural language)を記号列の体系として計算機に与えるだけでは，計算機は自然言語の意味を処理できない。そこで，意味情報を計算機で扱う場合，自然言語を中間言語に変換して扱うと都合が良い。中間言語は，自然言語の種類に依存しない意味記述の言語である。計算機は，中間言語から得られる自然言語の構造，及び意味情報をもとに種々の処理が可能である。しかし，中間言語をどのように定式化するかは非常に難しい。自然言語処理の研究において，この問題は重要である。

近年，著者等の研究グループは自然言語概念の意味表現形式の一つとしてSD式(Semantic-structure Description Form)を提案している[1,2,3,4]。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等を記述するための一種の中間言語であり，その構文は，曖昧さの無い文脈自由文法で規定されている[4]。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おうとするモデルをSD式意味モデルと言う。SD式意味モデルの最大の特徴は，従来からの意味記述モデルでは扱い難かった2つの概念間の意味的な近さを定量的に扱える点である。そのため，概念

上の二項関係として「詳述関係 (elaboration relation)」を定義し，これに基づく意味差を導入している。詳述関係は，SD式の構文構造に基づくものと知識データに基づくものとを定義している。ここでいう詳述関係とは，2つの概念D_i，D_jに関して，D_iの意味をより具体化したものがD_jであり，逆にD_jの意味をより抽象化したものがD_iであるような関係のことである。

SD式意味モデルが有効であるか否かは，SD式データを用いて具体的な処理を試してみなければならない。そのためにSD式意味モデルの実験システムSDENV-3 (SD-form ENVironment system version 3) を試作してきた[5,6,7]。SDENV-3は，SD式意味モデルのサブシステムを統合したシステムである。

これまでのSD式意味モデルに関する研究では，システムをPrologでプログラミングし，実験してきた。これは，SD式意味モデルを提案する際にPrologでプログラミングすることを考慮していたためでもある。しかし，Prologはオブジェクト指向言語ではないため複数人で複数のシステムを作成した場合，それぞれのプログラムを組み合わせて応用的なシステムとすることは非常に難しい。また，インターフェースが限られているために，画像や動画などのマルチメディアを用いたシステムを開発することも困難である。さらに、これまでにPrologで作成したシステムは，開発したPrologインタープリタ以外のPrologインタープリタでは記述文法の仕様が異なっているために，そのままでは動作せず，修正する場合にも多くの労力を要した。

本研究の目的は，これまでPrologで作成してきたSD式意味モデル実験システムSDENVを，プログラミング言語Perlで再構築し，Web上でも実験ができるように改良することである。Perlはテキスト処理に優れ，HTML言語と組み合わせることにより多様な表現を可能とする。プログラミングの際は，各種の処理をサブルーチンとして記述し，他のシステムに組み込めるようにすることとした。以下，Perlでプログラミングした SDENV を SDENV-4 と呼ぶ。

2章では，SD式意味モデルの概要を述べる。3章では，SD式意味モデル実験システムSDENV-4の概要を，4章では，SDENV-4の実験例を示す。最後に，5章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

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みである。このモデルに従った意味記述をSD式と呼ぶ。

以下では，SD式意味モデルの概要を述べる。ただし，実験システムSDENV-4で実験が可能な範囲に限る。

2.1 SD式の構文的分類

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区切り記号などの記号から構成される記号列である。SD式の構文は，以下の8種類に分類している[3]。

(1) 変数概念ラベル（変数ラベル）

X, X1, X2, ..., Y, ..., Z, ...

(2) 単純概念ラベル（単純ラベル）

時計，フランス，トム，買う，赤，65，$

(3) パラメータ付きラベル

・ドル(100)　：100ドル
・km(25)　：25km
・車($)　：車（“ラベル($)”は関係代名詞的用法における先行詞）

(4) 修飾形式

記号“/”を修飾子と呼び，この記号の右側で左側を修飾する。複数のSD式で並列的に修飾する場合は，結合子“*para*”を用いる。

・本/童話　：童話の本
・女性/(美しい)*para*(若い)　：若くて美しい女性
・車($)/[*s*(太郎), *v*(運転), *o*($)]　：太郎が運転する車

(5) 規定子形式

SD式に特別な役割を与えるために，以下の7種類の規定子を用いる。

・*nega*(D)：概念Dの否定
・*pass*(D)：概念Dを受け身概念にする
・*assu*(D)：概念Dの仮定
・*only*(D)：概念Dの限定
・*even*(D)：事実・極端な事例などの概念Dを強調する
・*ethr*(D)：概念Dのうちのどちらかであることを表す
・*fcus*(D)：概念Dが記述の焦点であることを表す

規定子を用いたSD式の形式を規定子形式という。規定子形式の例を以下に示す。

<Ex. 2-1>

・*nega*(売る)　：売らない（否定）
・*pass*(捨てる)　：捨てられる（受け身）
・*assu*([*s*(理恵), *v*(来る)])　：もし理恵が来れば（仮定）
・*only*(歩行者)　：歩行者だけ（限定）

(6) 結合子形式

2つの概念を何らかの関係で結合して新しい概念を作るものを結合子と呼び，その形式を結合子形式と呼ぶ。SDENV-4では，36種類の結合子を準備している。

addi, *andx*, *asas*, *asif*, *bcau*, *caus*, *csof*, *defi*, *devi*, *effe*, *eorx*, *equa*
excl, *exmp*, *incl*, *indu*, *inst*, *isa*, *kdof*, *like*, *mnus*, *mtpl*, *nseq*, *ofal*
oppo, *orxx*, *para*, *plus*, *prof*, *pseq*, *ptof*, *siml*, *summ*, *than*, *thru*, *vaof*

結合子形式の例を以下に示す。

〈Ex. 2-2〉

・(老人) *plus* (子供)　　：老人と子供
・(北京) *equa* (首都/中国)　：北京は中国の首都
・(量) *than* (質)　　：質より量

(7) 陳述形式

SD 式の陳述形式の構文は，英語の 5 文型を模している。また，受け身形の文を表す構文を文型 6，文型 7，文型 8，文型 9 としている。

陳述形式の中で用いる陳述機能項目は，次の通りである。

s：主語項目　　v：述語項目　　o：目的語項目
i：間接目的語項目　　c：補語項目　　b：行為者項目

陳述形式の例を以下に示す。

〈Ex. 2-3〉

(文型 1)：[$s(D_1)$, $v(D_2)$]
・[s(自分), v(テニス/時/毎日)]
：私は毎日テニスをする。
(文型 2)：[$s(D_1)$, $v(D_2)$, $c(D_3)$]
・[s(トム), v(である), c(アメリカ人)]
：トムはアメリカ人である。
(文型 3)：[$s(D_1)$, $v(D_2)$, $o(D_3)$]
・[s(娘), v(読む/相/完了), o(本/歴史)]
：娘は歴史の本を読み終えた。
(文型 4)：[$s(D_1)$, $v(D_2)$, $i(D_3)$, $o(D_4)$]
・[s(理恵), v(与える/過去), i(犬), o(ビスケット)]
：理恵は犬にビスケットを与えた。
(文型 5)：[$s(D_1)$, $v(D_2)$, $o(D_3)$, $c(D_4)$]
・[s(彼), v(名付ける/過去), o(娘/所有/彼), c(花子)]
：彼は，彼の娘を花子と名付けた。
(文型 6)：[$s(D_1)$, v(*pass*(D_2)), $b(D_3)$]
・[s(窓/当該), v(*pass*(割る)), b(トム)]
：その窓は，トムによって割られた。
(文型 7)：[$s(D_1)$, v(*pass*(D_2)), $i(D_3)$, $b(D_4)$]
・[s(お金/指示), v(*pass*(与える/過去)), i(自分), b(母)]

：このお金は，母によって私に与えられた。

(文型8)：[s(D$_1$), v(*pass*(D$_2$)), o(D$_3$), b(D$_4$)]

・[s(我々), v(*pass*(提供)), o(情報/あらゆる), b(マスコミ)]

：我々は，マスコミよってあらゆる情報を提供される。

(文型9)：[s(D$_1$), v(*pass*(D$_2$)), c(D$_3$), b(D$_4$)]

・[s(娘/所有/彼), v(*pass*(名付ける/過去)), c(花子), b(彼)]

：彼の娘は，彼によって花子と名付けられた。

(8) 感情形式

感情SD式は，自然言語による感情的な発声や発話の状況の記述に用いる。感情SD式では，呼びかけ・応答・感嘆の3種類を定義している。

感情形式の中で用いる感情機能項目は，次の通りである。

a：呼びかけ項目　r：応答項目　e：感嘆項目

感情形式の例を以下に示す。

<Ex. 2-4>

・[a(ナンシー)]　：ナンシー（呼びかけ）

・[r(否定)]　：いいえ（応答）

・[e([s(犬/遠指示), v(大きい)])]　：あの犬は，なんて大きいんだ！（感嘆）

2.2　SD式の意味的情報量

SD式では，記号列の構造で何か固有の概念を表現しようとするだけでなく，その概念の意味量の大小も表すこととしている。そのため，SD式意味モデルでは，各SD式記号に意味素量を設定している。任意のSD式をDとするとき，その意味量を

si(D)＝n〔*semit*〕

と表す。また，その単位をsemitとしている。SD式意味モデル実験システムSDENV-4の場合は，意味素量を以下のように設定している。

(1)　変数ラベル　“X, Y, Z, ...”　：1〔*semit*〕

(2)　単純ラベル　“馬, CAT, ...”　：10〔*semit*〕

(3)　修飾子　“/”　：1〔*semit*〕

(4)　規定子　“*nega, only, assu, ...*”　：2〔*semit*〕

(5)　結合子　“*plus, incl, para, ...*”　：1〔*semit*〕

(6)　機能項目記号　“*s, v, c, o, ...*”　：1〔*semit*〕

(7)　区切り記号　“[ ]”　：1〔*semit*〕

(8)　区切り記号　“( )”, “,”　：0〔*semit*〕

SD式全体の意味量は，それらの総和となる。

SD式意味モデルでは，これらを定義するときの一般的指針は示しているが，意味量の値については，モデルの利用者が独自に定めて良いとしている。

〈Ex. 2-5〉

si(靴/新しい)＝21

si([s(美紀), v(である), c(学生/長崎純心大学)])＝45

si([s(相手), v(読む/*mood*/義務), o(本/当該)])＝67

si([e(感嘆/賞賛)])＝23

2.3 2概念間の詳述関係

2つの概念間の「詳述関係（elaboration relation)」は，SD式意味モデルにおける最も基本的な枠組みである[6]。2つの概念D_i，D_jに関して，D_iの意味をより具体化したものがD_jであり，逆に，D_jの意味をより抽象化したものがD_iであるとき，D_iとD_jには詳述関係があるという。このときのD_iをD_jの先祖，D_jをD_iの子孫と呼んでいる。D_iからD_jへの詳述関係を

$elab(D_i, D_j)=n$ または $elab(D_i, D_j, n)$

と表す。ただし，n（$0 \leqq n < \infty$）は詳述量で，詳述の詳しさの程度を表すものである。

SD式意味モデルにおける詳述関係には，D_i，D_jそのものの構造による「構文的詳述関係」とシステムが利用できる知識データに基づく「知識に基づく詳述関係」の2種類があり，それぞれ，

$elab_{synt}(D_i, D_j)=n$, $elab_{know}(D_i, D_j)=n$

と表す。一般的には，

$elab(D_i, D_j)=min\{elab_{synt}(D_i, D_j), elab_{know}(D_i, D_j)\}=n$

と定義している。ここで，上式で定義される詳述関係は1段の関係である。この関係が連結して，多段の詳述関係となることもある。

2.3.1 構文的詳述関係

2つのSD式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。

(1) D_1＝X（変数ラベル），D_2が一般のSD式のとき

$$elab_{synt}(D_1, D_2)=si(D_2)-1$$

(2) D_1がD_2の一部分となっているとき（修飾形式)。この条件は必要条件である。

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

D_1＝D_2のとき

$$elab_{synt}(D_1, D_2)=0$$

(3) D_1とD_2が同一の陳述形式同士または感情形式同士のとき

$$elab_{synt}([s(D_{1s}), v(D_{1v})], [s(D_{2s}), v(D_{2v})])$$
$$=elab(D_{1s}, D_{2s})+elab(D_{1v}, D_{2v})$$
$$elab_{synt}([s(D_{1s}), v(D_{1v}), c(D_{1c})], [s(D_{2s}), v(D_{2v}), c(D_{2c})])$$
$$=elab(D_{1s}, D_{2s})+elab(D_{1v}, D_{2v})+elab(D_{1c}, D_{2c})$$

以下同様。

部分的な概念同士の詳述関係（$elab(D_{1s},D_{2s})$ など）については，2.3.3項で述べる。

(4) 陳述形式のSD式 D_1 と D_2 において，記号列 D_1 が記号列 D_2 の一部であるとき

$$elab_{synt}([s(D_{1s}),\ v(D_{1v})],\ [s(D_{2s}),\ v(D_{2v}),\ c(D_{2c})])$$
$$= elab(D_{1s},\ D_{2s}) + elab(D_{1v},\ D_{2v}) + si(D_{2c}) + 1$$

（機能項目記号“c”を考慮して意味量を1プラスする。）

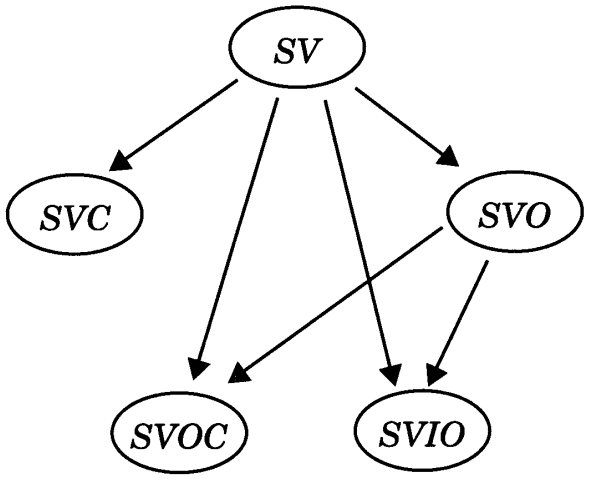
$$elab_{synt}([s(D_{1s}),\ v(D_{1v})],\ [s(D_{2s}),\ v(D_{2v}),\ i(D_{2i}),\ o(D_{2o})])$$
$$= elab(D_{1s},\ D_{2s}) + elab(D_{1v},\ D_{2v}) + si(D_{2i}) + si(D_{2o}) + 2$$

（機能項目記号“i”，“o”を考慮して意味量を2プラスする。）

以下同様。これらの関係を図1に示す。

(5) D_1 から D_2 への詳述関係が成り立たないとき

$$elab_{synt}(D_1,\ D_2) = \infty$$



S：$s(D_s)$
V：$v(D_v)$
C：$c(D_c)$
O：$o(D_o)$
I：$i(D_i)$
D_s, D_v, D_c, D_o, D_i, は任意のSD式。
矢印の始点が先祖（D_1）で終点が子孫（D_2）の関係である。

図1　陳述形式のSD式 D_1 と D_2 において，記号列 D_1 が記号列 D_2 の一部となる場合の関係

構文的詳述関係には，次のような性質がある。

(a) 構文的詳述関係は再帰的に定義される。

(b) D_1 から D_2 への構文的な詳述関係を繋ぎ合わせた「多段の構文的詳述関係」は，常に1段の構文的詳述関係とみなされる。すなわち，D_1 から D_2 への詳述関係は，D_1 と D_2 の構造から一意的に定まる。

（例）　$elab_{synt}$(壷, 壷/貴重) = 11

$elab_{synt}$(壷/貴重, 壷/(貴重) *para* (高価)) = 11

であれば，この関係は次のようにまとめられる。

$elab_{synt}$(壷, 壷/(貴重) *para* (高価)) = 22

(c) $elab_{synt}$ は D_1，D_2 の両方を明示的に与えた場合にのみ計算できる。

(d) $elab_{synt}(D_1,\ D_2)$ が最も小さな値（0以外）となるのは，

D_1 = X（変数ラベル），D_2 = 単純ラベル

の場合である。

構文的詳述関係の例を以下に示す。

〈Ex. 2-6〉

$elab_{synt}$(靴, 靴/新しい)＝si(靴/新しい)－si(靴)＝21－10＝11

$elab_{synt}$([s(美紀), v(である), c(学生)], [s(美紀), v(である), c(学生/長崎純心大学)])
＝11

2.3.2 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付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を，

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

と表す。知識に基づく詳述関係には，次の2種類がある。

(1) 個別の知識に基づく詳述関係

(2) 一般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，システム中に存在する個々の知識により求める。一方，一般の知識に基づく詳述関係は，経験上一般に成り立つと考えられる規則により定義している。知識に基づく詳述関係が成り立たないときは，

$elab_{know}(D_1, D_2)=\infty$

としている。

知識に基づく詳述関係は，システムに与えられた知識データから，そのシステムで$elab_{know}(D_1, D_2)$が成り立つ全ての概念対を探し出せる(片方，または両方が与えられていなくても詳述関係を計算できる）という性質がある。

(1) 個別の知識に基づく詳述関係

個別の知識とは，概念同士の関係を表す結合形式のSD式で記述されたものである。また，個別の知識に基づく詳述関係とは，次のような知識がシステムに与えられている場合の詳述関係である。

sk1：(D_1) *equa* (D_2)　または，(D_2) *equa* (D_1)　(D_1とD_2は等しい)
→ $elab_{know}(D_1, D_2)=0$

sk2：(D_1) *defi* (D_2) (D_1はD_2と定義されている)
→ $elab_{know}(D_1, D_2)=0$

sk3：(D_1) *csof* ([D_{21}, D_{22}, ..., D_{2i}, ...])　(D_1はD_{21}, D_{22}, ..., D_{2i}, ...から構成される)
→ $elab_{know}(D_1, D_2)=2$

sk4：(*assu* (D_2)) *caus* (D_1)　(もしD_2ならば，D_1である)
→ $elab_{know}(D_1, D_2)=2$

sk5：(*assu* ((D_2) *andx* (D_3))) *caus* (D_1)　(もしD_2かつD_3ならば，D_2である)
→ $elab_{know}(D_1, D_2)=2$

sk6：(D_1) *incl* (D_2)　(D_1はD_2を含む)

→ $elab_{know}(D_1, D_2)=3$

sk7：$(D_2)\,ptof\,(D_1)$　　(D_2はD_1の一部である)

→ $elab_{know}(D_1, D_2)=3$

sk8：$(D_2)\,kdof\,(D_1)$　　(D_2はD_1の一種である)

→ $elab_{know}(D_1, D_2)=3$

個別の知識に基づく詳述関係の例を以下に示す。

<Ex. 2-7>

システムに次の知識が定義されていることを前提とする。

(果物) *incl* (すいか)　：果物はすいかを含む。

(沖縄) *ptof* (日本)　：沖縄は日本の一部である。

(*assu*([*s*(X), *v*(である), *c*(教師/数学)])) *caus*([*s*(X), *v*(得意), *c*(数学)])

：もしXが数学の教師ならば，Xは数学が得意である。

この場合の個別の知識に基づく詳述関係と詳述量は次のようになる。

$elab_{know}$(果物, すいか)＝3

$elab_{know}$(日本, 沖縄)＝3

$elab_{know}$([*s*(次郎), *v*(得意), *c*(数学)], [*s*(山下(先生)), *v*(である), *c*(教師/数学)])＝2

(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，次のような知識に基づく詳述関係である。但し，D_nは任意のSD式である。

gk1：$elab_{know}(D, nega(nega(D)))=2$

gk2：$elab_{know}(D, fcus(D))=2$　　$elab_{know}(D, only(D))=2$

$elab_{know}(D, even(D))=2$　　$elab_{know}(D, ethr(D))=2$

gk3：$D_1=nega([s(D_a), v(D_b), ..., c(D_c), ...])$,

$D_2=[s(D_a), v(D_b), ..., c(nega(D_c)), ...]$ の場合

$elab_{know}(D_1, D_2)=2$

gk4：$D_1=nega([s(D_a), v(D_b), ...])$，$D_2=[s(D_a), v(nega(D_b)), ...]$ ならば

$elab_{know}(D_1, D_2)=2$

gk5：$D_1=[s(D_a), v(である), c(D_b)]$ の形式で，D_2が

$(D_a)\,defi\,(D_b)$，$(D_a)\,incl\,(D_b)$，$(D_b)\,equa\,(D_a)$，$(D_b)\,kdof\,(D_a)$，

のうちの何れかならば，

$elab_{know}(D_1, D_2)=2$

gk6：$D_1=assu(D_{10})$であり，かつ $elab(D_{10}, D_2)$ が成り立つならば，

$elab_{know}(D_1, D_2)=2$

さらに，D_i，D_j，D_kが名詞ラベルで，

$elab_{know}(D_i, D_j)=2$，$elab_{know}(D_i, D_k)=3$

を満たす場合，*gk7*～*gk18* の関係を定義している。

gk7 ：$elab_{know}$(D$_i$/X, D$_i$/SOME) = 1 （X：変数ラベル）

gk8 ：$elab_{know}$(D$_i$/SOME, D$_j$) = 1

gk9 ：$elab_{know}$(D$_i$/SOME, D$_k$) = 1

gk10：$elab_{know}$(D$_i$/SOME, D$_i$/MOST) = 3

gk11：$elab_{know}$(D$_i$/SOME, D$_i$/ANY) = 4

gk12：$elab_{know}$(D$_i$/SOME, D$_i$/D) = 2

gk13：$elab_{know}$(D$_i$/MOST, D$_i$/ANY) = 1

gk14：$elab_{know}$(D$_i$/D, D$_i$/ANY) = 2

gk15：$elab_{know}$(D$_j$/D, D$_i$/ANY) = 2

gk16：$elab_{know}$(D$_k$/D, D$_i$/ANY) = 2

gk17：$elab_{know}$(D$_j$, D$_i$/ANY) = 3

gk18：$elab_{know}$(D$_k$, D$_i$/ANY) = 3

gk7～*gk18* の関係を図 2 に示す。

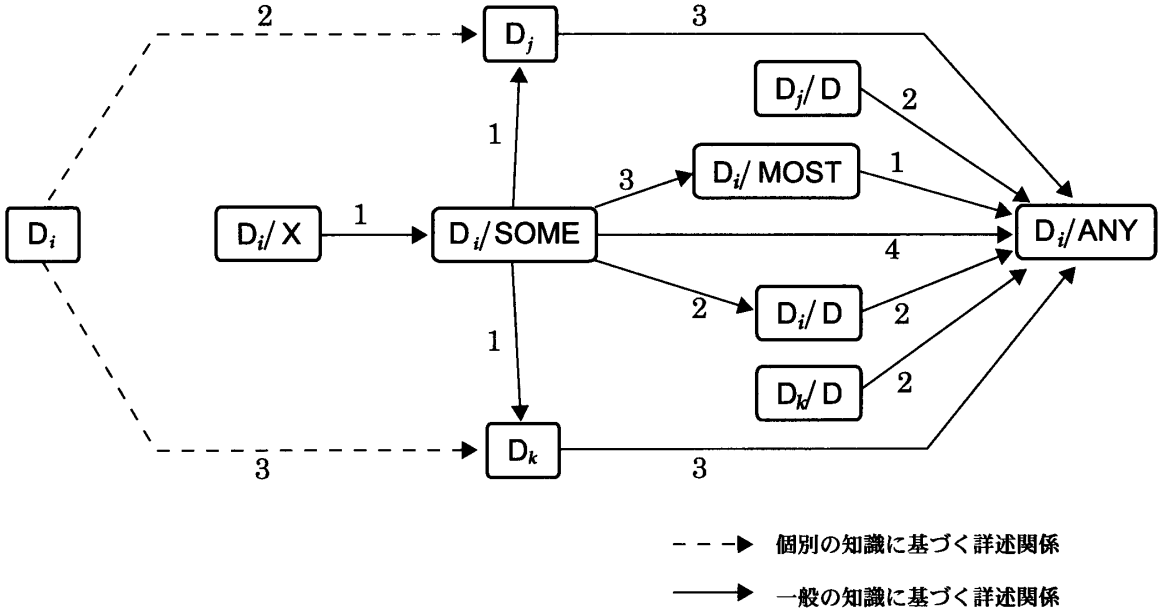


図 2　一般の知識に基づく詳述関係 *gk7*～*gk18* の模式図

一般の知識に基づく詳述関係の例を以下に示す。

〈Ex. 2-8〉

システムに次の知識が定義されていることを前提とする。

(人間) *csof* ([男性, 女性]) ：人間は男性と女性からなる。

(学生) *incl* (大学生) ：学生は大学生を含む。

このとき，個別の知識に基づく詳述関係により

$elab_{know}$(人間, 男性) = 2，$elab_{know}$(人間, 女性) = 2

であるので，

$elab_{know}$(人間/SOME, 男性) = 1 （∵ *gk8*）

$elab_{know}$(女性/美しい, 人間/ANY) = 2 （∵ *gk15*）

などが成り立つ。また，

$elab_{know}$(学生, 大学生) = 3

より，

$elab_{know}$(学生/SOME, 大学生) = 1 （∵ *gk9*）

$elab_{know}$(大学生, 学生/ANY)＝3　　(∵ *gk18*)

などが成り立つ。

2.3.3 一般的な詳述関係

前述のように，詳述関係には「構文的詳述関係」と「知識に基づく詳述関係」がある。もし構文的にも知識によっても詳述関係が成立するならば，次のように定義する。

$$elab(D_1, D_2) = min\{elab_{synt}(D_1, D_2),\ elab_{know}(D_1, D_2)\} \quad (2\text{-}1)$$

また，構文的詳述関係と知識に基づく詳述関係の2種類を連結して，複雑な詳述関係を表すことができる。m 個の詳述関係を連結してできる詳述関係を「m 段の詳述関係」といい，

$$elab_m(D_1, D_2) = n,\ elab_m(D_1, D_2, n) \quad (n：詳述量) \quad (2\text{-}2)$$

と表す。m 段の詳述量 n は，次式により求める。

$$n = min\{elab_M(D_1, D_2)\} \quad (M = 1, 2, 3, ..., m) \quad (2\text{-}3)$$

(2-1) 式で定義される詳述関係は，1段の詳述関係である。また，m 段の詳述関係において2種類の詳述関係をどのように連結したかということを詳述関係の経路という。2.3.1項で述べたように，多段の構文的詳述関係は，1段の詳述関係とみなせる。ゆえに，m 段の詳述関係における経路の総数は，2^m以下である(m 段の経路数はフィボナッチ数に従う：$C_m = C_{m-1} + C_{m-2}$)。

以下に m 段($m = 1, 2, 3$)の詳述関係の全経路を示す。ただし，S は構文的詳述関係を，K は知識に基づく詳述関係を表す。

1段：D_1-(K)-D_2,　　D_1-(S)-D_2

2段：D_1-(K)-(K)-D_2,　　D_1-(K)-(S)-D_2,　　D_1-(S)-(K)-D_2

3段：D_1-(K)-(K)-(K)-D_2,　　D_1-(K)-(K)-(S)-D_2,

D_1-(K)-(S)-(K)-D_2,　　D_1-(S)-(K)-(K)-D_2,

D_1-(S)-(K)-(S)-D_2

現実のシステムでは，詳述関係の段数 m を或る有限の値 (N) 以下に限定しなければ詳述量を求めるアルゴリズムは終了しない。いわゆる3段論法ができるためには $2 \leqq N$ が必要であるが，現実には計算量の点で $N < 5$ 程度のシステムしか実現できないであろう。

構文的詳述関係については，与えられた2つの概念を構成するそれぞれの部分同士の詳述関係に分解できることがある (2.3.1項参照)。例えば，D_1，D_2が何れも同じ形をした陳述形式の場合は次のようになる。

$$elab_{synt}([s(D_{11}), v(D_{12}), o(D_{13})], [s(D_{21}), v(D_{22}), o(D_{23})])$$
$$= elab(D_{11}, D_{21}) + elab(D_{12}, D_{22}) + elab(D_{13}, D_{23})$$

ただし，右辺の *elab* は(2-2)式で定義されるものである。ゆえに，一般の詳述関係は，再帰的な関係である。

3．SD 式意味モデル実験システム SDENV-4

SD 式意味モデルが有効であるか否かは，実際の SD 式データを用いて具体的に処理を試してみる必要がある。そこで本研究では，SD 式意味モデルを Web 上で実験するためのシステムとして SDENV-4（SD-form ENVironment system version 4）を試作した。

SD 式意味モデル実験システム SDENV-4 は，Perl version 5.6.1 でプログラミングしている。SDENV-4 では，SD 式意味モデルの応用システムでも利用できるように，それぞれの処理をサブルーチンとして記述している。他のシステムで SDENV-4 を利用するには，Perl の require 関数を用いてサブルーチン群（付録 1 を参照）を記述したファイル“sdenv4.pl”を取り込む。Perl は，テキスト処理に優れ，Web 上で実験ができるようにプログラミングすることができ，HTML 言語と組み合わせることにより多様な表現を可能とする。SDENV-4 は，次の URL で公開しており，ブラウザ上で動作実験ができるようにしている。

http://www.n-junshin.ac.jp/shota/sdenv4.cgi

Web 上で SDENV-4 を動作させるために必要なファイルは次の 4 つで，“sdenv4.cgi”から“sdenv4.pl”，“fact.dat”と“sd.log”を取り込み，処理を制御している。

sdenv4.cgi ：Web 上での動作実験用 SDENV-4 メインルーチン
sdenv4.pl ：SDENV-4 のサブルーチン群
fact.dat ：知識データ
sd.log ：Web 上での動作実験結果のログファイル

SDENV-4の主な機能を以下に示す。

(1) 入力 SD 式の形式をチェック (2.1節参照)
(2) 意味量の計算 (2.2節参照)
(3) 構文的詳述関係 (2.3.1項参照)
(4) 知識に基づく詳述関係 (2.3.2項参照)
(5) 一般の詳述関係 (2.3.3項参照)
(6) 知識データの更新機能（登録・削除）

SDENV-4 の入力ページを図 3 に示す。

SDENV-4 で実験する際には，図 3 において，まず，実験したい処理「sd（入力 SD 式の形式をチェック）」，「si（意味量の計算）」，「elab（意味量の計算）」，「elabsynt（構文的詳述関係）」，「elabknow（知識に基づく詳述関係）」の何れかを選択する。次に“Input”の欄に SD 式を入力する。このとき，入力の途中では改行しない。また，2 つ以上の SD 式を引数として入力する際の区切りは“半角スペース”を使用する。入力後に“OK”ボタンを押すと処理の結果がページの下方に表示される。結果の履歴数は，8 としている。

「elab」または「elabknow」について実験する際には，必要に応じて知識データを登録・削除する必要がある。図 3 の左上のメニュー［Knowledge Regist］を選択すると，図 4 のペー

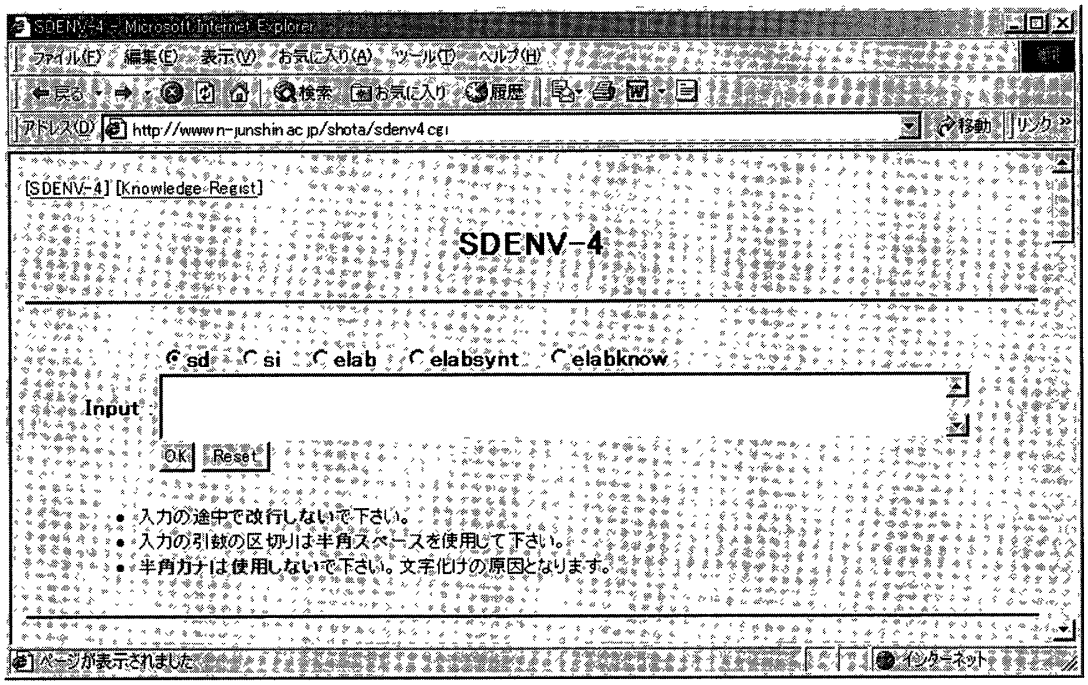


図 3　SDENV-4 の入力ページ

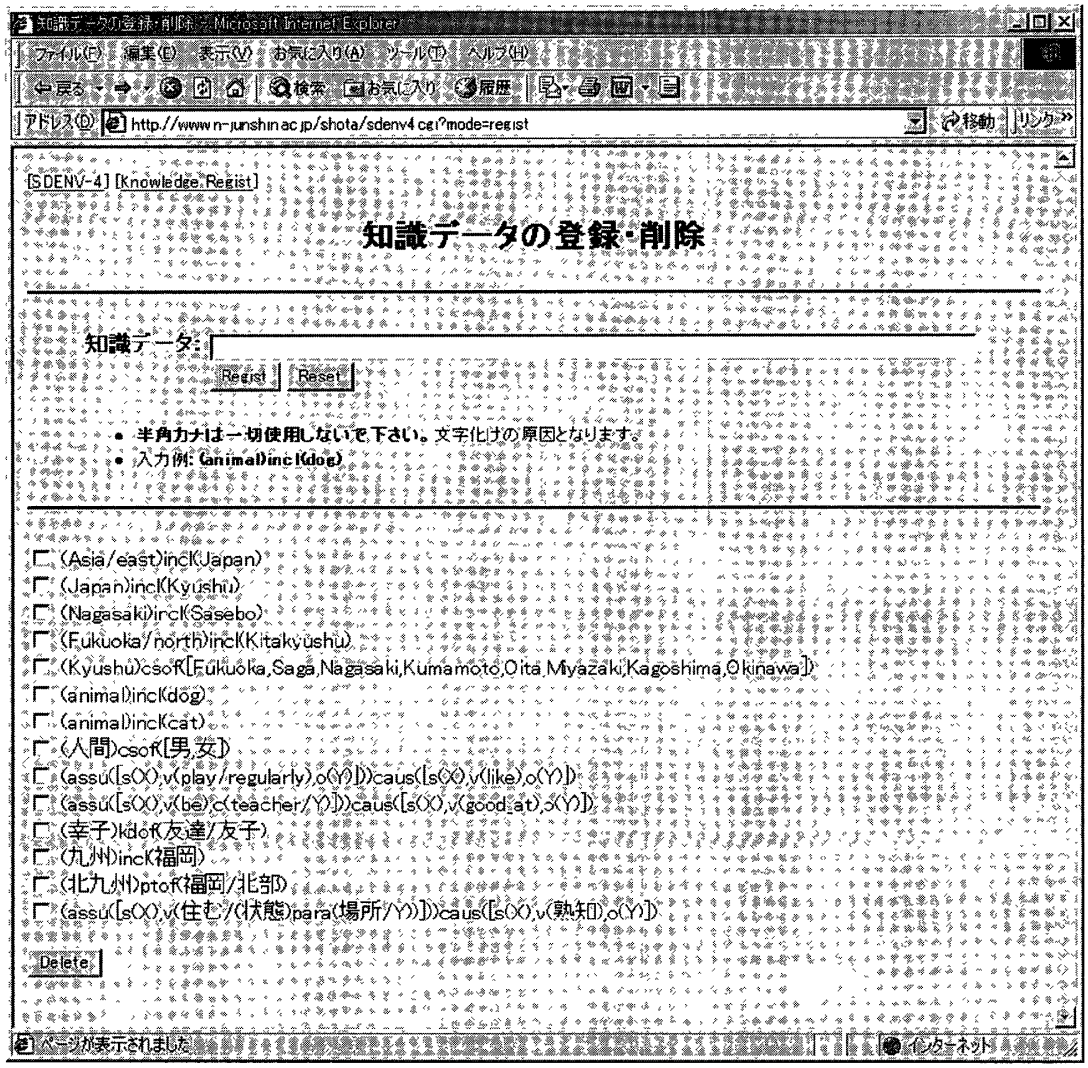


図 4　SDENV-4 の「知識データ登録・削除」のページ

ジが表示される。図4において“知識データ”の欄にSD式で記述した知識データを入力して“OK”ボタンを押すと登録され，ページの下方に現在登録されている全ての知識データが表示される。知識データを削除したい場合には，削除したい知識データのチェックボックスをオンにし，“Delete”ボタンを押す。

4．実験例

第2章で示した例は，SDENV-4によってその妥当性を確かめたものである。以下では，更に「SD式の形式」，「意味量の計算」，「構文的詳述関係」，「知識に基づく詳述関係」，「一般の詳述関係」の例を示す。

4.1 SD式の形式チェック：「sd」

入力されたものがSD式であるか否かを調べ，SD式であればそのSD式の形式(2.1節参照)を結果として出力する。SD式の形式とそれに対応するSDENV-4の戻り値の関係を表1に示す。

表1 SD式の形式に対応するSDENV-4の戻り値

SD式の形式		値	SD式の形式		値
陳述形式	文型1	type1	感情形式	呼びかけ	type_a
	文型2	type2		応答	type_r
	文型3	type3		感嘆	type_e
	文型4	type4	結合子形式		connector
	文型5	type5	規定子形式		prescriptor
	文型6	type6	修飾形式		modify
	文型7	type7	変数ラベル		variable
	文型8	type8	単純ラベル		label
	文型9	type9	SD式の形式でないもの		no

SDENV-4の入力ページ（図3参照）において，処理の種類は「sd」を選択する。

以下にSD式の形式を調べる処理の例を示す。

〈Ex. 4-1〉

入力：[s(dog/(big)*para*(white)), v(drink/past), o(milk)]

（白くて大きい犬は，ミルクを飲んだ。）

結果：type 3

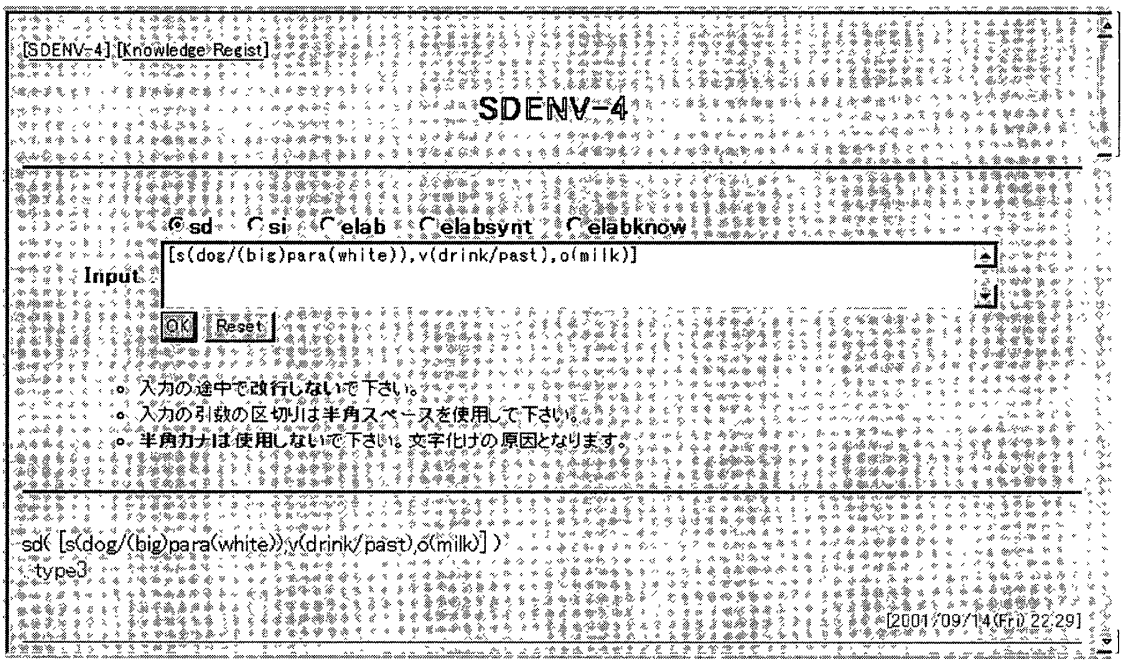


図 5　SD 式の形式チェックの例

4.2　意味量の計算：「si」

入力 SD 式の意味量(2.2節参照)を計算し，その結果を出力する。SDENV-4 の入力ページ(図 3 参照)において，処理の種類は「si」を選択する。

以下に SD 式の意味量計算の例を示す。

〈Ex. 4-2〉

入力：[*s*(dog/(big)*para*(white)), *v*(drink/past), *o*(milk)]

（白くて大きい犬は，ミルクを飲んだ。）

結果：67 [semit]

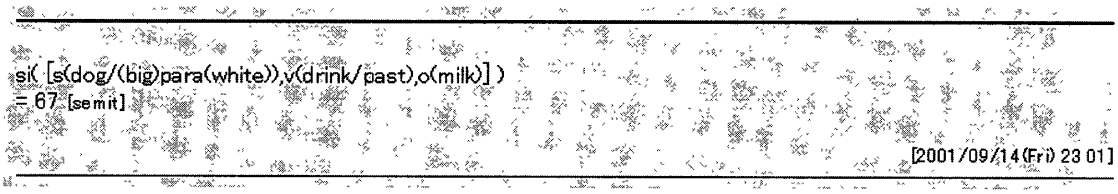


図 6　SD 式の意味量計算の例

4.3　構文的詳述関係：「elabsynt」

2 つの入力 SD 式 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か否かをチェックし，成り立つならばその関係の詳述量を結果とする。成り立たない場合の結果は，“∞”である（2.3.1項参照)。SDENV-4 の入力ページ（図 3 参照）において，処理の種類は「elabsynt」を選択する。入力の際には，D_1が D_2の先祖（D_2が D_1の子孫）の関係とした場合，D_1，D_2の順に 2 つの SD 式の間を半角スペースで区切って入力する。

以下に，構文的詳述関係の例を示す。

〈Ex. 4-3〉

入力：dog dog/(big) *para* (white)

結果：22 [semit]

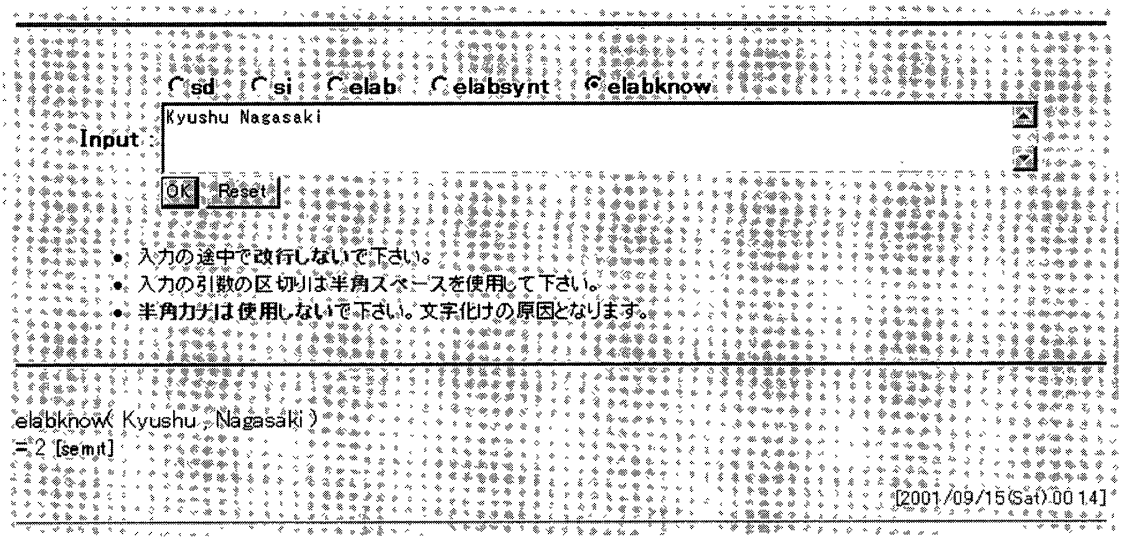


図 7　構文的詳述関係の例

4.4　知識に基づく詳述関係：「elabknow」

2 つの入力 SD 式 D_1，D_2について，構文的な詳述関係がない場合にも，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を生じる（2.3.2項参照）。SDENV-4 の入力ページ（図 3 参照）において，処理の種類は「elabknow」を選択する。D_1，D_2を明示的に与えた場合には，知識データを参照し，知識に基づく詳述関係が成り立つならば，その関係の詳述量を結果とする。成り立たない場合の結果は，“∞”である。入力の際には，D_1が D_2の先祖（D_2が D_1の子孫）の関係とした場合，D_1，D_2の順に 2 つの SD 式の間を半角スペースで区切って入力する。

知識に基づく詳述関係は，システムに与えられた知識データから，そのシステムで $elab_{know}(D_1, D_2)$ が成り立つ全ての概念対を探し出せる。この場合，D_1または D_2を変数（D, D1, D2, ..., X, X1, X2, ..., Y, Y1, ..., Z, Z1, ...）として入力する。

以下に，知識に基づく詳述関係の例を示す。

〈Ex. 4-4〉

システムに次の知識データが与えられていることを前提とする。

(Kyushu) *csof* ([Fukuoka, Saga, Nagasaki, Kumamoto, Oita, Miyazaki, Kagoshima, Okinawa])

（九州は，福岡，佐賀，長崎，熊本，大分，宮崎，鹿児島，沖縄で構成されている。）

(1)　D_1，D_2を明示的に与えた場合

入力：Kyushu Nagasaki

結果：2 [semit]

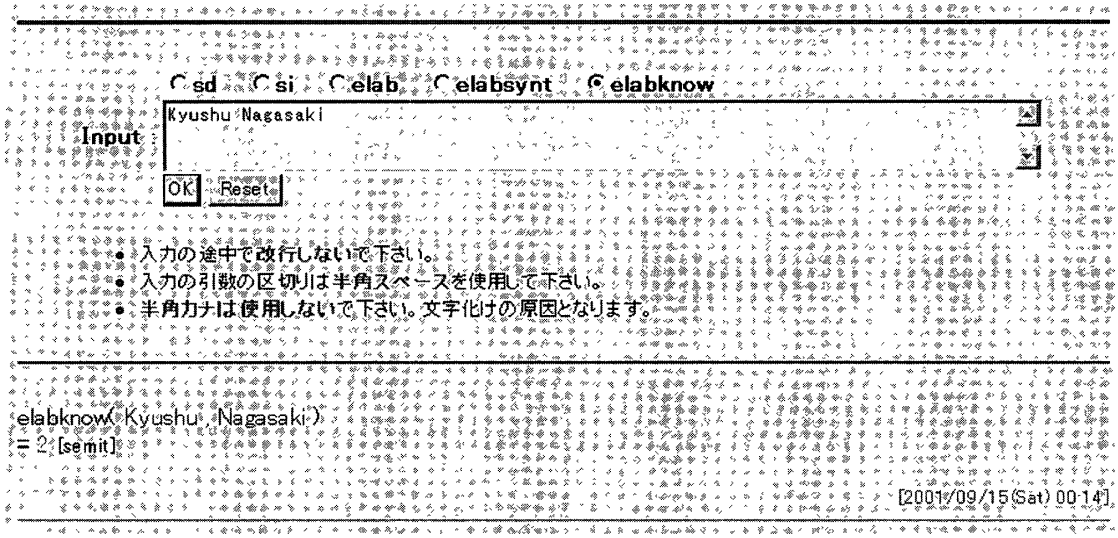


図 8　知識に基づく詳述関係の例（D_1，D_2を明示的に与えた場合）

(2)　D_1が変数の場合

入力：X Nagasaki

結果：$elab_{know}$(Kyushu, Nagasaki) $=2$ [semit]

$elab_{know}$(Kyushu/some, Nagasaki) $=1$ [semit]

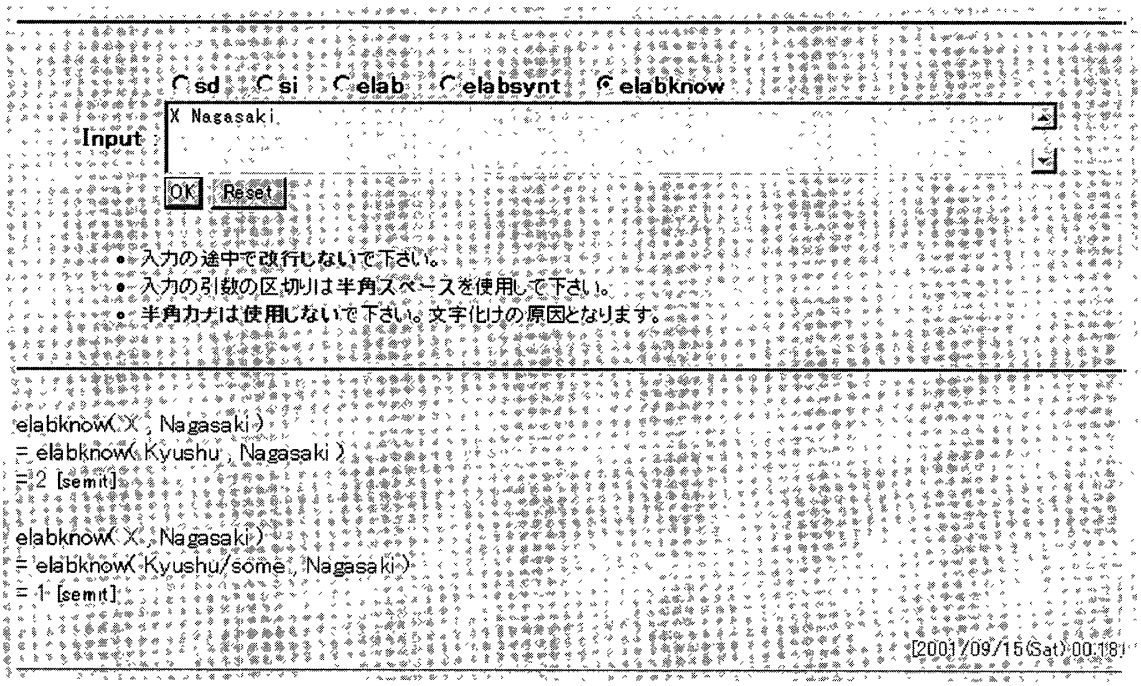


図 9　知識に基づく詳述関係の例（D_1が変数の場合）

4.5　一般の詳述関係：「elab」

詳述関係は，「構文的詳述関係」と「知識に基づく詳述関係」がある(2.3.3項参照)。もし，構文的にも知識によっても詳述関係が成立するならば，

$elab(D_i, D_j) = min\{elab_{synt}(D_i, D_j), elab_{know}(D_i, D_j)\} = n$

と定義している。また，m 個の詳述関係を連結して m 段の詳述関係が成り立つ場合がある。SDENV-4 では，3段の詳述関係までを処理する。SDENV-4 の入力ページ（図3参照）において，処理の種類は「elab」を選択する。入力に関しては「知識に基づく詳述関係 elabknow」と同様である。

以下に，詳述関係の例を示す。

〈Ex. 4-5〉

システムに次の知識データが与えられていることを前提とする。

(理恵) *kdof* (友達/詩織)　　：理恵は詩織の友達の一人である。

(九州) *incl* (福岡)　　：九州は福岡を含む。

(北九州) *ptof* (福岡/北部)　　：北九州は福岡北部の一部である。

(*assu*([*s*(X), *v*(住む/(状態)*para*(場所/Y))])) *caus*([*s*(X), *v*(熟知), *o*(Y)])

：もし X が Y に住んでいるならば，X は Y を熟知している。

このとき，

elab([*s*(友達/詩織), *v*(熟知), *o*(九州)], [*s*(理恵), *v*(住む/(状態)*para*(場所/北九州))]) = 22

となる。図10は，この詳述関係を示したものである。全体としては，2段の詳述関係であるが，部分的には3段の詳述関係を含んでいる。図10の S は構文的詳述関係，K は知識に基づく詳述関係である。

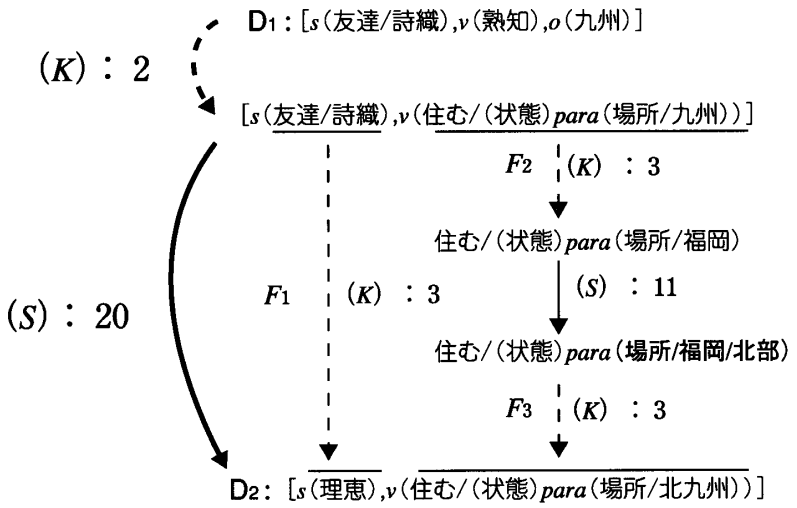


図10　詳述関係

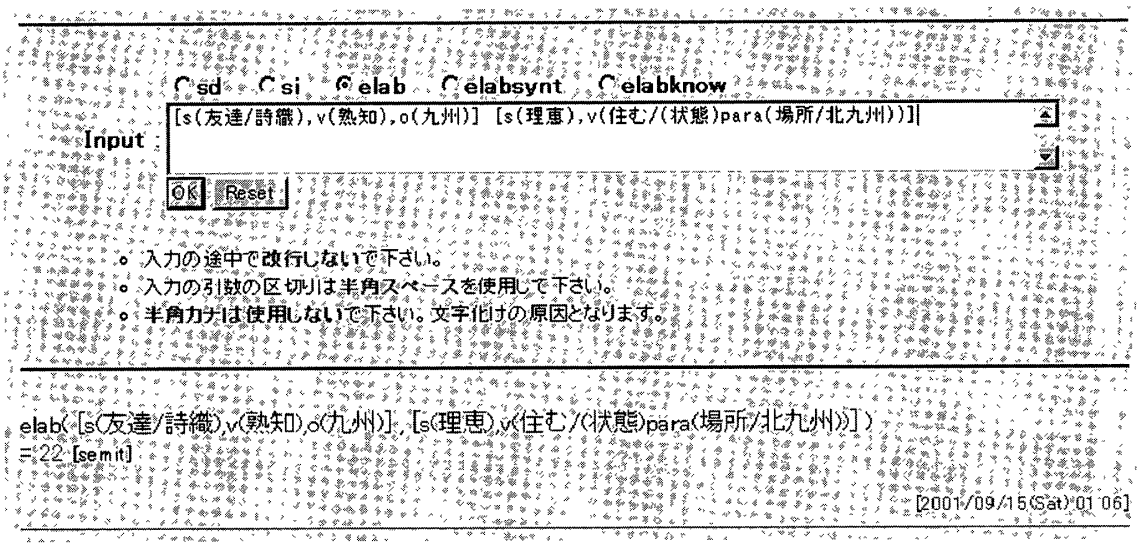


図11 詳述関係の例

5．おわりに

SD式意味モデルの有効性を確認するための実験システムSDENV-4を，プログラミング言語Perlを用いて試作した。従来のSDENVおよびSD式意味モデルに関する研究では，Prologでシステムを開発してきた[5,6,7]。しかし，Prologはオブジェクト指向言語ではないため複数人で複数のシステムを作成した場合，それぞれのプログラムを組み合わせて応用的なシステムとすることは非常に難しい。また，インターフェースが限られているために，画像や動画などのマルチメディアを用いたシステムを開発することも困難である。そこで，Prologに替えてPerlでSDENVを再構築することにした。Perlは，テキスト処理に優れており，Web上で実験ができるようにプログラミングすることができる。また，HTML言語と組み合わせることにより多様な表現も可能である。プログラミングの際は，各種の処理をサブルーチンとして記述し，他のシステムに組み込めるようにした。

今回試作したSDENV-4では，SD式意味モデルにおける「入力SD式の形式の種類をチェック」，「意味量の計算」，「構文的詳述関係」，「知識に基づく詳述関係」，「一般の詳述関係」を求める機能がある。また，知識データをWeb上で容易に登録・削除する機能も付加した。以上の機能については，SD式意味モデル実験システムとして役割を果たしているが，SD式意味モデルで提案している内容全てを網羅しているわけではない。今後の課題は，SDENV-4に「最近共通先祖の導出」，「意味差の計算」の機能をつけることである。SDENVをPerlでプログラミングする場合，Prologに標準で備えられているパターンマッチの機能等もPerlでプログラミングしなければならず，多くの労力を要する側面もあるが，上述した長所により，今後のSD式意味モデル応用システムでSDENV-4が有効的に利用されることが期待できる。

参 考 文 献

[1] Kawaguchi, E., Wakiyama, M. and Nozaki,K.: "A Semantic Structure Description Model of General Concepts in Natural Language World", Proc. of PRICAI, pp.298-303 (1990)

[2] Kawaguchi, E., Kamata, S. and Wakiyama, M.: "Elaboration Relation and the Nearest Common Ancestor of a Concept Pair in the SD-Form Semantics Model", Proc. of 2nd PRICAI, pp.426-432 (1992)

[3] Kawaguchi, E., Kamata, S. and Wakiyama, M.: "The Semantic Metric Computation Scheme in the SD-Form Semantics Model", Proc. of 3rd PRICAI, pp.623-629 (1994)

[4] Wakiyama, M., Shao, G. and Kawaguchi, E.: "The Toward Generalization of the Semantics Metric in the SD-Form Semantics Model", Proc. of 4th PRICAI '96 Poster, pp.61-68 (1996)

[5] 吉原将太,脇山正博,河口英二："SD式意味モデル実験システムの試作"，情報処理学会九州支部研究会報告，pp.247-254 (1999)

[6] 吉原将太,脇山正博,河口英二："SD式意味モデルにおける3概念からの帰納推論システムの試作"，純心女子短期大学「紀要」第37集，pp.23-39 (2000)

[7] Shouta Yoshihara, Masahiro Wakiyama and Eiji Kawaguchi: "An Experimental System of Inductive Inference from a Concept-Triple in SD-Form Semantics Model", In: H. Jaakkola et al. (Eds.), INFORMATION MODELLING AND KNOWLEDGE BASES XI, IOS Press & Ohmsha, pp. 169-176 (2001)

付録1　SDENV-4の主要なサブルーチンの種類と機能

&sdform_check(D)
：DがSD式であるかどうかを調べる
［戻り値］
文型1 (SV)："type1"，
文型2 (SVC)："type2"，
文型3 (SVO)："type3"，
文型4 (SVIO)："type4"，
文型5 (SVOC)："type5"，
文型6 (SVB)："type6"，
文型7 (SVIB)："type7"，
文型8 (SVOB)："type8"，
文型9 (SVCB)："type9"，
呼びかけ(A)："type_a"，
応答(R)："type_r"，感嘆(E)："type_e"，
結合子形式："connector"，
規定子形式："prescriptor"，
修飾形式："modify"，変数ラベル："variable"，
単純ラベル："label"，上記以外："no"

&sd_type(D)
：陳述SD式・感情SD式の文型を調べる処理
［戻り値］
文型1 (SV)："type1"，
文型2 (SVC)："type2"，
文型3 (SVO)："type3"，
文型4 (SVIO)："type4"，
文型5 (SVOC)："type5"，
文型6 (SVB)："type6"，
文型7 (SVIB)："type7"，
文型8 (SVOB)："type8"，
文型9 (SVCB)："type9"，
呼びかけ(A)："type_a"，
応答(R)："type_r"，感嘆(E)："type_e"，
上記文型以外："no"

&sd_prescriptor(D)
：規定子形式であるか否かをチェック
［戻り値］真："prescriptor"，偽："no"

&prescriptor_type(D)
：規定子形式の種類をチェック
(例) &prescriptor_type("*only*(Taro)")
［戻り値］真："only"，偽："no"
(戻り値の種類：nega，pass，assu，only，even，ethr，fcus)

&sd_connector(D)
：結合子形式であるか否かをチェック
［戻り値］真："connector"，偽："no"

&connector_type(D)
：結合子形式の種類をチェック

(例) &connector_type("(Japan) *incl* (Tokyo)")
［戻り値］真："incl"，偽："no"
（戻り値の種類：para，addi，ndx，sas，asif，bcau，aus，csof，defi，devi，effe，eorx，equa，excl，exmp，incl，indu，inst，isa，kdof，like，mnus，mtpl，nseq，ofal，oppo，orxx，plus，prof，pseq，ptof，siml，summ，than，thru，vaof）

&sd_labe(D)
：単純ラベルであるか否かをチェック
(例) &sd_labe("dog")
［戻り値］真："label"，偽："no"

&sd_variable(D)
：変数ラベルであるか否かをチェック
(例) &sd_variable("X")
［戻り値］真："variable"，偽："no"

&sd_modify(D)
：修飾形式であるか否かをチェック
(例) &sd_variable("dog/(big) *para* (white)")
［戻り値］真："modify"，偽："no"

&si(D)
：SD式の意味量の計算
(例) &si(dog/(big) *para* (white))
［戻り値］32

&si_type(D,N)
：陳述形式・感情形式のSD式の意味量を求める
（D：SD式，N：これまでの計算過程で求めた意味量）
(例) &si_type([*s* (dog), *v* (drink), *o* (milk)], 0)
［戻り値］23

&si_connector(D, N)
：結合子形式のSD式の意味量を求める
（D：SD式，N：これまでの計算過程で求めた意味量）
(例) si_connector((kyushu) *incl* (fukuoka), 0)
［戻り値］21

&si_prescriptor(D, N)
：規定子形式のSD式の意味量を求める
（D：SD式，N：これまでの計算過程で求めた意味量）
(例) si_prescriptor(*nega* (drink/past), 0)
［戻り値］23

&si_modify(D,N)
：規定子形式のSD式の意味量を求める
（D：SD式，N：これまでの計算過程で求めた意味量）
(例) si_modify(dog/(big) *para* (white)), 0)
［戻り値］32

&elab(D1, D2)
：一般の詳述関係．構文的にも知識によっても詳述関係が成立する場合．ただし，3段までの詳述関係。
(例) &elab(animal, dog)
［戻り値］3
（fact. datに（animal) *incl* (dog) の知識が登録されている場合）

&elab1(D1, D2)
：1段の詳述関係。
(例) &elab1(animal, dog)
［戻り値］3（偽の場合："∞"）
（fact. datに（animal) *incl* (dog) の知識が登録されている場合）

&elab2(D1, D2)
：2段の詳述関係
(例) &elab2(Asia, Kyushu)
［戻り値］6（偽の場合："∞"）
（fact. datに（Asia) *incl* (Japan)，
（Japan) *incl* (Kyushu) の知識が登録されている場合）

&elab3(D1, D2)
：3段の詳述関係
(例) &elab3(Asia, Nagasaki)
［戻り値］9（偽の場合："∞"）
（fact. datに（Asia) *incl* (Japan)，
（Japan) *incl* (Kyushu)，
（Kyushu) *incl* (Nagasaki) の知識が登録されている場合）

&elab_x_d(X, D2)
：第1引数が変数の場合の詳述関係。ただし，3段までの詳述関係。
(例) &elab_x_d(X, dog)
［戻り値］真：animal，3　偽：no，∞
（fact. datに（animal) *incl* (dog) 知識が登録されている場合）

&elab_d_x(D1, X)
：第2引数が変数の場合の詳述関係。ただし，2段までの詳述関係。
(例) &elab_d_x(animal, X)
［戻り値］真：dog，3　偽：no，∞
（fact. datに（animal) *incl* (dog) の知識が登録されている場合）

&elab1_x_d(X,D2)
：1段の詳述関係　—第1引数が変数—
(例) &elab1_x_d(X, dog)
［戻り値］真：animal，3　偽：no，∞
(fact. dat に (animal) *incl* (dog) の知識が登録されている場合)

&elab1_d_x(D1, X)
：1段の詳述関係　—第2引数が変数—
(例) &elab2_d_x(animal, X)
［戻り値］真：dog，3　偽：no，∞
(fact. dat に (animal) *incl* (dog) の知識が登録されている場合)

&elab2_x_d(X, D2)
：2段の詳述関係　—第1引数が変数—
(例) &elab2_x_d(X, Kyushu)
［戻り値］真：Asia，6　偽：no，∞
(fact. dat に (Asia) *incl* (Japan)，
(Japan) *incl* (Kyushu) の知識が登録されている場合)

&elab2_d_x(D1, X)
：2段の詳述関係　—第1引数が変数—
(例) &elab2_d_x(Asia, X)
［戻り値］真：Kyushu，6　偽：no，∞
(fact. dat に (Asia) *incl* (Japan)，
(Japan) *incl* (Kyushu) の知識が登録されている場合)

&elab3_x_d(X, D2)
：3段の詳述関係　—第1引数が変数—
(例) &elab3_x_d(X, Nagasaki)
［戻り値］真：Asia，9　偽：no，∞
(fact. dat に (Asia) *incl* (Japan)，
(Japan) *incl* (Kyushu)，
(Japan) *incl* (Nagasaki) の知識が登録されている場合)

&elab3_d_x(D1, X)
：3段の詳述関係　—第2引数が変数—
(例) &elab3_d_x(Asia, X)
［戻り値］真：Nagasaki，9　偽：no，∞
(fact. dat に (Asia) *incl* (Japan)，
(Japan) *incl* (Kyushu)，
(Japan) *incl* (Nagasaki) の知識が登録されている場合)

&elabsynt(D1, D2)
：2概念間の構文的詳述関係。D1, D2は明示的に与えること。
(例) &elabsynt(dog, dog/big)
［戻り値］真：11　偽：∞

&elabknow(D1, D2)
：2概念間の知識に基づく詳述関係
(例) &elabknow(Kyushu, Nagasaki)
［戻り値］真：3　偽：∞
(fact. dat に (Kyushu) *incl* (Nagasaki) の知識が登録されている場合)

&elabk_x_d(X,D2)
：知識に基づく詳述関係　—第1引数が変数—
(例) &elabk_x_d(X, Nagasaki)
［戻り値］真：Kyushu，3　偽：no，∞
(fact. dat に (Kyushu) *incl* (Nagasaki) の知識が登録されている場合)

&elabk_d_x(D1, X)
：知識に基づく詳述関係　—第2引数が変数—
(例) &elabk_d_x(Kyushu, X)
［戻り値］真：Nagasaki，3　偽：no，∞
(fact. dat に (Kyushu) *incl* (Nagasaki) の知識が登録されている場合)

&sd_split(D)
：陳述形式・感情形式のSD式を分割
(例) &sd_split([*s*(dog), *v*(drink), *o*(milk)])
［戻り値］配列　s(dog)，v(drink)，o(milk)
陳述形式・感情形式以外 “no”

&sd_modi_split(D)
：修飾形式のSD式を "/", "para" で分割
(例1) &sd_modi_split(dog/(big) *para* (white))
［戻り値］配列　dog，big，white
(例2) 分割できない場合
&sd_modi_split(dog)
［戻り値］dog

&label_pickup(D)
：陳述形式・感情形式の各要素の概念を取り出す処理
(例) &label_pickup(“*s*(dog)”)
［戻り値］dog
(例) ピックアップできない場合
&label_pickup(“dog”)
［戻り値］dog

&replace_assu_caus1(D1, KD1, KD2)
：(assu(KD1)) *caus* (KD2) 内の変数に対して対応するラベルに置換

(例) &replace_assu_caus1([s(太郎), v(得意), c(英語)], [s(X), v(である), c(先生/Y)], [s(X), v(得意), o(Y)])
[戻り値] 真：[s(太郎), v(である), o(先生/英語)]
偽："no"

&replace_assu_caus2(DI, KDI, KD2)
：(assu(KD1))caus(KD2)内の変数に対して対応するラベルに置換
(例) &replace_assu_caus2([s(太郎), v(である), c(先生/英語)], [s(X), v(である), c(先生/Y)], [s(X), v(得意), o(Y)])
[戻り値] 真：[s(太郎), v(得意), o(英語)]
偽："no"

&good_result1(@arg)
：配列の中で重複した解を削除して返す
(例) &good_result1(@arg)
@arg＝animal, 3, animal/some, 1, animal, 3
[戻り値] animal, 3, animal/some, 1