知能と複雑系　132－2
(2003. 3. 13)

論理プログラムにおける説明推論過程の視覚化システムの実装

吉田　聡†　笹倉万里子††

† 岡山大学大学院 自然科学研究科　〒700–8530 岡山市津島中 3–1–1
†† 岡山大学 工学部　〒700–8530 岡山市津島中 3–1–1
E-mail: †{yoshi,sasakura}@momo.it.okayama-u.ac.jp

あらまし　本論文では拡張論理プログラムにおける説明推論過程を視覚化する方法を提案する．説明推論で推移するゴールと導出を呼び出す関係を記憶し，その視覚化を行う．その記憶する処理を従来の説明推論手続き [1] に加えたものが，本論文で述べる視覚化機能付き説明推論手続きである．導出は木構造で呼び出すので，それぞれの導出のゴール履歴を紙のようなものに表現し，それを 3 次元上に配置する．この方法で視覚化を行うシステムの実装を行う．
キーワード　視覚化，推論

A visualization of reasoning procedures for a logic program and its implementation

Satoshi YOSHIDA† and Mariko SASAKURA††

† Graduate School of Natural Science and Technology, Okayama University　Tushimanaka 3–1–1, Okayama, 700–8530 Japan
†† Faculty of Engineering, Okayama University　Tushimanaka 3–1–1, Okayama, 700–8530 Japan
E-mail: †{yoshi,sasakura}@momo.it.okayama-u.ac.jp

Abstract　In this paper, we propose a procedure that visualizes a reasoning process for an extended logic program. The procedure is an extension of the abductive procedure[1]. We record histories of goals and derivation calls in the procedure, then visualize them. Since derivation calls forms a tree, we display it in 3-dimension of which a node is represented as a paper that shows a history of goals in each derivation.
Key words　visualization, reasoning

1. はじめに

コンピュータを用いた視覚化の研究は，計算機で処理する大量のデータをわかりやすく表示するために，CG(コンピュータグラフィックス) を用いて表示する目的で始まった．そして計算機の性能の向上や低価格化により様々な分野への応用が可能となった．流体力学や高分子化学で膨大な量の科学的データを CG を用いて表示することを科学的視覚化という．また計算機ユーザインタフェイス，ソフトウェア開発など様々な分野で，対話手段としてグラフィックスが利用されるようにもなった．このように抽象的な情報を視覚化することを，科学的視覚化に対して情報視覚化と呼ぶ [2]．本研究は情報視覚化のひとつである．

情報視覚化の目的は，人間にとってわかりにくい情報を見ただけで直観的に理解できるように表示することである．本研究でも扱うオイラー図は集合の概念を説明するとき必ず用いられる．これは記号だけでは新しい概念を説明することが難しいためで，何かを説明するときにたいてい人は図を使って思考の手助けをする．図を使うことは新しい概念を理解・構築する時に有用だということは経験的に知られている．情報視覚化はその経験則に基づく．

例えば見えログ [3] はただのテキストから特徴情報を取り出し，それを視覚化することでユーザがログ情報を調査することを支援する．またプログラムビジュアリゼーションという分野ではプログラムのある側面や実行状態を表示するために視覚化を用いる．本研究は，論理プログラムにおける導出過程を理解しやすいものとするのが目的で，プログラムビジュアリゼーションのデータ視覚化システム [4] のひとつである．動的なデータと静的なデータの両方で視覚化を行う．ここで言う導出とは説明推論手続き [1] を用いて推論を行うこととする．

本論文では拡張論理プログラムにおける導出過程を視覚化する方法を提案する．視覚化するのは導出の結果と過程である．

導出過程に関してはSLD導出 [1] と失敗による否定がある場合の導出の呼び出しという過程の2つを表現する．まず導出を行うとゴールが推移する．その推移を導出の流れと捉え，オイラー図を拡張した図で表現する．この図で手続きの再帰性を表現している．失敗による否定がある場合に導出を呼び出すが，この場合は別の図で表現する．それぞれの導出をそれぞれの図で表現することで図の複雑さを解消できる．そしてこの図の円に色をつけることで，導出が成功したなどの導出結果を表す．

またこのままではそれぞれの図の関係，つまり導出が導出を呼び出した関係がわかりにくいと考え，この図を紙のようなものとして3次元上に配置する．配置した位置関係や線で結ぶことによりそれぞれの図の関係も視覚化する．つまり3次元上で表現された導出を見ることで，どんな感じでゴールが変遷していったのかを表現する．それは円の数や色，紙のようなものの位置や数で表している．

本研究では提案する方法に基づき導出過程の視覚化システムの実装を行う．本システムは論理プログラムをルールとして与え，ゴールが示されると自動的に導出を行い，その導出過程を動的に視覚化を行う．また導出が終了した時点で導出過程をよりわかりやすく視覚化するために，導出過程を3次元で表示する．

本論文では2節で拡張論理プログラムにおける説明推論手続きを紹介し，3節で視覚化手続きを示し，4節でシステムについて述べ，5節で例を示し，最後に6節で結論を述べる．

2. 拡張論理プログラムにおける説明推論手続き

本研究における基礎事項の説明を参考文献 [1] に従って行う．

2.1 拡張論理プログラム

拡張論理プログラム [1] は2種類の否定を用いることで「Aでない」(明示的否定) 以外の「Aであるとは言えないので，Aでない」(失敗による否定) という命題を自然に記述できる．

拡張論理プログラムは

$$L \leftarrow L_1, \ldots, L_m, {\sim}L_{m+1}, \ldots, {\sim}L_n (1 \leqq m \leqq n)$$

という形の節の集合である．L や $L_i (1 \leqq i \leqq n)$ はリテラル，すなわちアトムまたはアトムの明示的否定，そして "$\sim$" は失敗による否定を表す．L を頭部，$L_1, \ldots, L_m, {\sim}L_{m+1}, \ldots, {\sim}L_n$ を本体という．もし $L = A$ ならば $\neg L$ は $\neg A$ を意味し，$L = \neg A$ ならば $\neg L$ は A を意味する．A と $\neg A$ は対で相補的であるという．

ゴール (goal) は

$$\leftarrow L_1, \ldots, L_m, {\sim}L_{m+1}, \ldots, {\sim}L_n (1 \leqq m \leqq n)$$

という表現であり，$L_i (1 \leqq i \leqq n)$ はリテラルである．リテラルが含まれていなければ，$\square$ によって表示する．

2.2 説明推論手続き

説明推論手続きは，結果とルールから原因を推論する手続きである．説明推論は与えられたゴールに対し，成功手続きと失敗手続きを再帰的に用いながら導出を行う．つまり結果をゴール，ルールを論理プログラムとして，導出がどのように行われたかという原因を推論していく．詳細については参考文献 [1] に示されている．

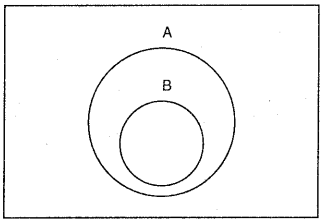

図1　オイラー図による論理プログラム節の視覚化

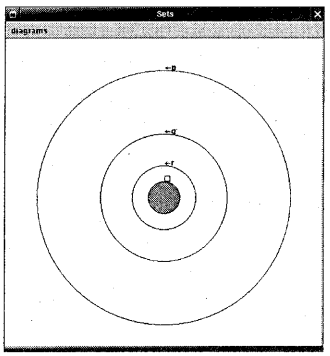

図2　P_1 において $\leftarrow p$ をゴールとしたときに導出の視覚化を行った例

導出に対して，導出過程をわかりやすく表示するために視覚化を行う手続きを組み込んだものが，視覚化機能付き説明推論手続きである．3節で視覚化機能付き推論手続きと視覚化のルールと方法を示す．

3. 視覚化手続き

3.1 視覚化手法の概要

本論文で提案する視覚化法はオイラー図を基としている [5][6]．オイラー図は集合の概念を表現するためによく使われる図だが，推論ルールを表現することもできる．例えば次の論理プログラムがあるとする．

$A \leftarrow B$

これは $A \vee \neg B$ とも書ける．$\mathbf{A}$ は A が真である場合に全部の集合で，$\mathbf{B}$ は B が真である場合に全部の集合であるとするとこの論理プログラムは

$\mathbf{A} \supseteq \mathbf{B}$

と同値であるので図1のオイラー図で表現できる．この考え方を拡張したものが本論文の視覚化法である．

ある論理プログラムをルールにしてゴールを与えて導出を行うと，ゴールが変わっていく．例えば論理プログラム

$$P_1 = \{\ p \leftarrow q$$
$$q \leftarrow r$$
$$r \leftarrow \square\ \}$$

において，p をゴールとして導出を行っていくと $G_0 = \leftarrow p, G_1 = \leftarrow q, G_2 = \leftarrow r, G_3 = \square$ となり，成功導出が成功する．G_j については3.2節で示す．p が正しいことを調べるために，q が正しいことを調べるという流れでゴールが変わる．このゴールの変遷をオイラー図を基に，視覚化したものが図2であ

る．図は r が成功することにより，q が成功し，q が成功することにより p が成功することを示している．これが本論文で行う論理プログラムにおける導出過程の視覚化の基本的な形である．

視覚化機能付き説明推論手続きを 3.2 節で示す．3.3 節でその健全性について触れる．そして 3.4 節で導出過程の視覚化について述べる．

3.2 視覚化機能付き説明推論手続き

成功導出および失敗導出では拡張論理プログラム P の中の節を用いてゴールを変換していく．その変換はゴールの中の 1 つのリテラルもしくはリテラルの失敗による否定に対して以下の操作を行うことである．ゴールを $G =\leftarrow M_1, ..., M_m$ とする．$M_i(1 \leqq i \leqq m)$ を選んだとき l をリテラルとして，

(1) $M_i = l$: M_i が頭部である節を探し，節があればその本体で M_i を置き換える．

(2) $M_i =\sim l$: 成功導出の場合は l に関する失敗導出を呼ぶ．失敗導出の場合は l に関する成功導出を呼ぶ．これは成功導出の場合は l が失敗するということで $\sim l$ が成功することを導きたいのに対し，失敗導出の場合は l が成功することで $\sim l$ が失敗することを導きたいからである．

(3) $\leftarrow l$ と $\leftarrow \neg l$ が同時に成功することを防ぐために導出の過程で成功すると過程したリテラルを記憶する．記録するリテラルの集合を Σ で表す．

(4) 失敗による否定 $\sim l$ については，成功導出から失敗導出を呼び出すときに l を記憶する．記憶する場所を Δ とする．Δ を用いることで $\{p \leftarrow \sim q, q \leftarrow \sim p\}$ というプログラムに対し $\leftarrow p$ あるいは $\leftarrow q$ というゴールが与えられたときに答えを得ることができることが示されている [7]．l を Δ に記憶することはそのときに行っている導出が l の失敗を仮定していることを示す．

この手続きでは，ゴールとそのときの Σ, Δ を組としたものをノード，ゴールの変化をそのノード間の枝と考えた木で推論過程を表現し，その木を視覚化する．以下に視覚化機能付き推論手続きの成功導出，失敗導出の手続きを述べる．与えられた拡張論理プログラムを P とする．

成功導出を 3 字組の列で表現する．

$(G_0, \Sigma_0, \Delta_0), ..., (G_h, \Sigma_h, \Delta_h)$ または，

$(G_0, \Sigma_0, \Delta_0) \Rightarrow_{suc-P} (G_h, \Sigma_h, \Delta_h)$.

ここで，$G_i(0 \leqq i \leqq h)$ はゴールの変化を記憶するリテラルの集合，Σ_i は成功すると仮定したリテラルの集合，Δ_i は失敗導出を呼び出すときに使われたリテラルの集合である．また 3 字組の列を

$Abs_j = (G_j, \Sigma_j, \Delta_j)$.

と表現する．Abs_j は導出中の G_j, Σ_j, Δ_j の状態を示す．

成功導出においては，導出が停止し，$G_h = \square$ となれば終了と言う．そして終了する成功導出を

$(G_0, \Sigma_0, \Delta_0) \Rightarrow_{SUC-P} (G_h, \Sigma_h, \Delta_h)$.

と表記する．

失敗導出では与えられたゴールが失敗することを導けたとき，この導出が終了したという．失敗導出は成功導出と同様に次のように表現される．

$(H_0, \Sigma_0, \Delta_0), ..., (H_h, \Sigma_h, \Delta_h)$ または，

$(H_0, \Sigma_0, \Delta_0) \Rightarrow_{ff-P} (H_h, \Sigma_h, \Delta_h)$.

ここで，ゴールの集合 $H_i(0 \leqq i \leqq h)$ はゴールの変化を記憶するゴールの集合である．Σ_i, Δ_i は成功導出と同様である．また 3 字組の列を

$Abh_j = (H_j, \Sigma_j, \Delta_j)$.

と表現する．Abh_j は導出中の H_j, Σ_j, Δ_j の状態を示す．

失敗導出は，$H_h = \bullet$ となれば終了と言う．ただし $\bullet$ とはここでは導入する記号で以下の失敗導出手続きにおいて適用できるものがなかったときに導入する記号で，終了する失敗導出を

$(H_0, \Sigma_0, \Delta_0) \Rightarrow_{FF-P} (H_h, \Sigma_h, \Delta_h)$.

と表記する．これ以降，成功導出，失敗導出に関する記号 $\Rightarrow_{suc-P}, \Rightarrow_{SUC-P}, \Rightarrow_{ff-P}, \Rightarrow_{FF-P}$ でプログラムを意味する P を省略し，それぞれ $\Rightarrow_{suc}, \Rightarrow_{SUC}, \Rightarrow_{ff}, \Rightarrow_{FF}$ と表記する．

説明推論手続きは成功導出と失敗導出を再帰的に呼び出しながら導出を行う手続きである．成功導出が導出を呼び出したときに今までの導出を視覚化してある図から別の図に移り，また新たに導出の視覚化を行う．なぜなら同じ図で全てを表示するよりも，見た目が単純になるからである．そこで 1 枚の図に表示させる円の情報と，導出を呼び出した図と呼び出された図の関係が重要になる．

1 枚の図に表示させる円は，導出が進むことによって変遷しているゴールである．視覚化するにはこのゴールであるリテラルの集合を変遷する順で記憶しておく必要がある．またそれぞれの図の関係は図の履歴という形で記憶する．導出が導出に呼び出されたとき，どの導出から呼び出されたかを記憶しておくことで履歴がわかる．

視覚化する情報を格納しておく場所を以下のように表記する．

$(WG_0, WP_0), ..., (WG_w, WP_w)$.

WG_w は 1 つの図で表示するゴールであるゴール集合のリストである．また WP_w は図の履歴を格納するリストである．本論文では WG_w, WP_w を以下のように記述する．

$WG_w = [wg_0, wg_1, ..., wg_f, ..., wg_g]$.

$WP_w = [wp_0, wp_1, ..., wp_o, ..., wp_p]$.

各要素 $wg_f(0 \leqq f \leqq g)$ はゴールの集合，$wp_o(0 \leqq o \leqq p)$ は整数である．WG_w, WP_w に視覚化を行う情報を格納する手続きを説明推論手続きに加えたものが，視覚化機能付き説明推論手続きである．以下に手続きを示す．

視覚化機能付き成功導出手続き

導出を w 番目に呼び出された導出とする．この成功導出を呼び出した導出が v 番目 $(0 \leqq v < w)$ に呼び出されたものであるとき，その視覚化情報 WP_v の最後尾に整数 w を格納したものが WP_w である．すなわち $WP_v = [wp_0, wp_1, ..., wp_o]$ であるとすると，

$WP_w = [wp_0, wp_1, ..., wp_o, w]$.

ただし $WP_0 = [0]$ である．

説明推論手続きでは成功導出は必ずただ 1 つのリテラルまたはリテラルの失敗による否定からなるゴール $G_0 =\leftarrow M_1$ (M_1 はリテラルまたはリテラルの失敗による否定) で始まる．したがって，まず G_0 を WG_w に格納する．このとき $WG_w = []$ なので $wg_0 = \{G_0\}$ となる．

$WG_w = [\{G_0\}]$.

そのあと, $Abs_k(0 < k)$ で $G_k = \leftarrow M_1, ..., M_m (M_j, 1 \leqq j \leqq m$ はリテラルまたはリテラルの失敗による否定) であるとき, $M_j (1 \leqq j \leqq m)$ を選択する. そして (S1) か (S2) を導出が停止するまで行う.

(S1) $G_k \neq \square$ **のとき**

(s1) $M_j = l$**(ただし** l **はリテラル) のとき**

$\neg l \notin \Sigma_k$ で, $l \leftarrow L_1, ..., L_n$(ここで $L_1, ..., L_n$ はリテラルまたはリテラルの失敗による否定) という節が P の中にある場合,

$G_{k+1} = \leftarrow M_1, ..., M_{j-1}, L_1, ..., L_n, M_{j+1}, ..., M_m$

$\Sigma_{k+1} = \Sigma_k \cup \{M_j\}$, $\Delta_{k+1} = \Delta_k$

$Abs_{k+1} = (G_{k+1}, \Sigma_{k+1}, \Delta_{k+1})$

$WG_w = [\{G_0\}, \{G_1\}, ..., \{G_k\}, \{G_{k+1}\}]$

(s2) $M_j = \sim l$**(ただし** l **はリテラルのとき)**

(s2-1) $l \in \Delta_k$ ならば,

$G_{k+1} = \leftarrow M_1, ..., M_{j-1}, M_{j+1}, ..., M_m$

$\Sigma_{k+1} = \Sigma_k$, $\Delta_{k+1} = \Delta_k$, $Abs_{k+1} = (G_{k+1}, \Sigma_{k+1}, \Delta_{k+1})$

$WG_w = [\{G_0\}, \{G_1\}, ..., \{G_k\}, \{G_{k+1}\}]$

(s2-2) $l \notin \Delta_k$ で失敗導出 $(\leftarrow l, \Sigma_k, \Delta_k \cup \{l\}) \Rightarrow_{FF} (H', \Sigma', \Delta')$ があれば,

$G_{k+1} = \leftarrow M_1, ..., M_{j-1}, M_{j+1}, ..., M_m$

$\Sigma_{k+1} = \Sigma'$, $\Delta_{k+1} = \Delta'$, $Abs_{k+1} = (G_{k+1}, \Sigma_{k+1}, \Delta_{k+1})$

$WG_w = [\{G_0\}, \{G_1\}, ..., \{G_k\}, \{G_{k+1}\}]$

(S2) $G_k = \square$ **のとき**

導出が終了する.

視覚化機能付き失敗導出手続き

導出を w 番目に呼び出された導出とする. この失敗導出を呼び出した導出が v 番目 $(0 \leqq v < w)$ に呼び出されたものであるとき, その視覚化情報 $WP_v (0 \leqq v < w)$ の最後尾に整数 w を格納したものが WP_w である. すなわち, $WP_v = [wp_0, wp_1, ..., wp_o]$ であるとすると,

$WP_w = [wp_0, wp_1, ..., wp_o, w]$

失敗導出は $H_0 = \leftarrow M_1 (M_1$ はリテラルまたはリテラルの失敗による否定) で始まるので, H_0 を WG_w に格納する. このとき $WG_w = []$ なので $wg_0 = H_0$ となる.

$WG_w = [H_0]$.

そのあと, $Abh_k (0 < k)$ で $H_k = \{\leftarrow M_1, ..., M_m\} \cup H'_k (M_j, 1 \leqq j \leqq m$ はリテラルまたはリテラルの失敗による否定) である $M_j (1 \leqq j \leqq m)$ を選択する. そして (F1) か (F2) を導出が停止するまで行う.

(F1) $H_k \neq \bullet$ **のとき**

(f1) $M_j = l$ **(ただし** l **はリテラル) のとき**

(f1-1) $(\leftarrow \neg l, \Sigma_k, \Delta_k) \Rightarrow_{SUC} (G', \Sigma', \Delta')$ であれば,

$H_{k+1} = \bullet$

$\Sigma_{k+1} = \Sigma'$, $\Delta_{k+1} = \Delta'$, $Abh_{k+1} = (H_{k+1}, \Sigma_{k+1}, \Delta_{k+1})$

$WG_w = [H_0, H_1, ..., H_k, H_{k+1}]$

(f1-2) l で, $l \leftarrow L_1, ..., L_n$(ここで $L_1, ..., L_n$ はリテラルまたはリテラルの失敗による否定) という節が P の中に r 個 $(1 \leqq r)$ ある場合に, すべての節の本体の集合を $RL = \{l_1, ..., l_q, ..., l_r\}$ とするとき,

(f1-2-1) $r \geqq 2$ で, かつすべての $l_q (1 \leqq q \leqq r)$ に対して, 失敗導出 $(\leftarrow l_q, \Sigma_k, \Delta_k) \Rightarrow_{FF} (H', \Sigma', \Delta')$ があるとき,

$H_{k+1} = \{\leftarrow M_1, ..., M_{j-1}, L_1, ..., L_n, M_{j+1}, ..., M_m\} \cup H'_k$

$H_{k+2} = \bullet$

$\Sigma_{k+1} = \Sigma'$, $\Delta_{k+1} = \Delta'$, $Abh_{k+1} = (H_{k+2}, \Sigma_{k+1}, \Delta_{k+1})$

$WG_w = [H_0, H_1, ..., H_k, H_{k+1}, H_{k+2}]$

(f1-2-2) $r = 1$ のとき,

$H_{k+1} = \{\leftarrow M_1, ..., M_{j-1}, L_1, ..., L_n, M_{j+1}, ..., M_m\} \cup H'_k$

$\Sigma_{k+1} = \Sigma_k$, $\Delta_{k+1} = \Delta_k$, $Abh_{k+1} = (H_{k+1}, \Sigma_{k+1}, \Delta_{k+1})$

$WG_w = [H_0, H_1, ..., H_k, H_{k+1}]$

(f1-3) l を左辺に持つ節がない場合

$H_{k+1} = \bullet$

$\Sigma_{k+1} = \Sigma_k$, $\Delta_{k+1} = \Delta_k$, $Abh_{k+1} = (H_{k+1}, \Sigma_{k+1}, \Delta_{k+1})$

$WG_w = [H_0, H_1, ..., H_k, H_{k+1}]$

(f2) $M_j = \sim l$**(ただし** l **はリテラル) のとき**

(f2-1) $l \in \Delta_k$ ならば,

$H_{k+1} = \{\leftarrow M_1, ..., M_{j-1}, M_{j+1}, ..., M_m\} \cup H'_k$

$\Sigma_{k+1} = \Sigma_k$, $\Delta_{k+1} = \Delta_k$, $Abh_{k+1} = (H_{k+1}, \Sigma_{k+1}, \Delta_{k+1})$

$WG_w = [H_0, H_1, ..., H_k, H_{k+1}]$

(f2-2) $l \notin \Delta_k$ で, $(\leftarrow l, \Sigma_k, \Delta_k) \Rightarrow_{SUC} (G', \Sigma', \Delta')$ があれば,

$H_{k+1} = \bullet$

$\Sigma_{k+1} = \Sigma'$, $\Delta_{k+1} = \Delta'$, $Abh_{k+1} = (H_{k+1}, \Sigma_{k+1}, \Delta_{k+1})$

$WG_w = [H_0, H_1, ..., H_k, H_{k+1}]$

(F2) $H_k = \bullet$ **のとき**

導出が終了する.

3.3 視覚化機能付き説明推論手続きの健全性

本論文の視覚化機能付き説明推論手続きは参考文献 [5] の手続きをもとに拡張したものである. 参考文献 [5] で説明推論手続きの健全性は証明されているので, 本論文の手続きが参考文献のものと同じ処理を行い, かつ視覚化機能の部分において, WG_w にゴールの履歴がもれなくすべて記憶されること, また WP_w に導出の呼び出し関係が正しく記録されることができれば, 本論文の視覚化機能付き説明推論手続きも健全である. 説明推論手続きと視覚化機能付き説明推論手続きの違いは 2 つある.

- 視覚化機能を加えた部分
- 視覚化機能付き失敗手続きの (f1-2)

この手続きの健全性は参考文献 [8] で示している.

3.4 導出過程の視覚化

3.1 節でオイラー図を拡張した導出過程の視覚化法の例を示した. 以下に本研究の視覚化方針を詳しく述べる. ここで, Abs_j と Abh_j は違うものではあるが, 視覚化方針は Abs_j と Abh_j の両方に対して, 同様の考え方や処理を用いるので Ab_j と表記することで Abs_j と Abh_j の両方を示していることとする.

- 長方形で区切られた内部を 1 つの図と呼ぶことにする.
- Ab_j は一つの円で表される. それを C_i と表記する.
- Ab_j と Ab_{j+1} は同じ図の中に描かれる. WG_w により Ab_j と Ab_{j+1} の関係がわかるので, Ab_{j+1} が Ab_j に呼び出されていた場合, C_j は C_{j+1} を含むように描く. C_i が C_j を含むとは C_j が C_{j+1} の輪郭の中に完全に入っていることを言う.
- Ab_j に呼び出された Ab_{j+1} が複数ある場合, それぞれの

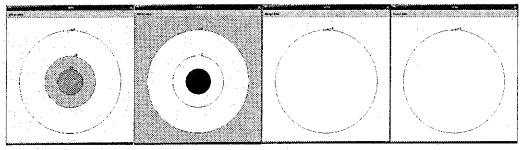

図 3　P_2 において $G_0 = \leftarrow A$ で導出の視覚化を行った例

Ab_j 以下の部分木を別な図として表示する.

- 図の 1 番外側にある円のリテラルをこの図の根と考える.

成功導出の図と失敗導出の図を区別するために, 失敗導出の図では背景を薄い灰色で塗りつぶす. また, 推論が成功したかどうかをわかりやすくするため, 次の方針で円に色をつけることにする.

- 成功導出で $G_j = \square$ なら Ab_j の円の中の色は灰色.
- 成功導出において Ab_{j+1} が Ab_j より失敗導出を呼び出すことによって得られた場合, および失敗導出において成功導出を呼び出すことによって得られた場合, Ab_j の円の中の色は薄い灰色.
- 失敗導出で $G_j = \bullet$ なら Ab_j の円の中の色は黒.
- 失敗導出で複数の部分木として, 円を複数描く. このとき円の中の色は黒.
- Ab_j 以下の部分木を別の図として表示する場合, Ab_{j-1} の円内に描かれる Ab_j の円の色は, その別の図の最も内側の円の色と同じにする.
- それ以外のとき円の中の色は白.

論理プログラム P_2 に対し, 視覚化を行ったものが図 3 である.

$$P_2 = \{\ A \leftarrow\sim B$$
$$B \leftarrow C\ \ \}$$

本論文で実現する導出過程の視覚化は, 以下の 2 つからなる.

- 導出中の導出過程の視覚化
- 導出後の導出過程の視覚化

3.4.1　導出中の導出過程の視覚化

例としてあげた図 3 は導出が終了したもので, 図 3 を見れば導出を全体的に見渡すことができるという意味でわかりやすい. 導出手続きを理解している人にとっては直観的に導出過程を理解する助けになるであろう. しかし導出手続きを理解していない人にとっては手続きを見ないと導出を理解することは難しい. 特に導出が導出を呼び出すタイミング, つまりどれがどの図を, いつ呼び出したかを理解するには時間がかかる. そこで導出中は動的に視覚化を行う. これは導出中にゴールが変わるとき視覚化する情報が更新される. ここで視覚化を行うウインドウも再描画する. これにより手続きを理解できていない人にとって, 手続きの動作が直観的にわかり, 手続きの理解を促すことができる. P_2 の導出に対して動的な視覚化を示したものが図 4 である. 右下の数字の順に表示される.

3.4.2　導出部分と視覚化部分の連動

本研究では導出を行う過程そのものを視覚化したいと考えた. その実現方法は, 3.4.1 節で説明した視覚化を動的に行うことで導出過程そのものの視覚化を実現する.

成功導出で G_j が G_{j+1} または失敗導出で H_j が H_{j+1} になるときに図に表示させる円の情報として G_{j+1} や H_{j+1} を得ている. そこで導出中, 新たに表示させる円の情報を WG_w に加えた瞬間に, 最新の WG_w の情報に基づき視覚化を更新することで, 現在の導出状況を表す. また更新されたら必ず数秒導出を中断した後に導出を続ける.

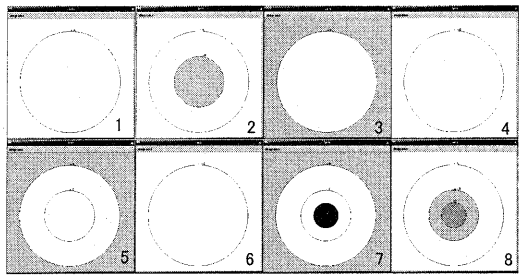

図 4　P_2 において $G_0 = \leftarrow A$ で導出を動的な視覚化を示した例

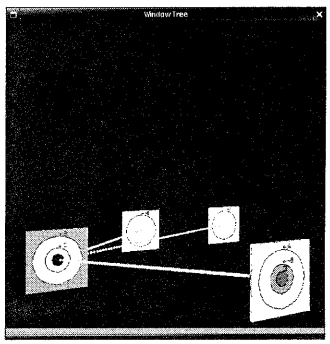

図 5　P_2 において $G_0 = \leftarrow A$ で導出過程を 3 次元で視覚化を行った例

3.4.3　導出終了後の視覚化

導出終了後は導出の図をそれぞれ見ることで導出過程を確認できるが, 複数の図が表示されると見た目が複雑になる. 導出を追うことで図の関係はわかるが, 導出を確認するのは面倒である. そこで図の関係も直観的にわかるよう視覚化を行う. それぞれの図を薄い紙として捉え, 3 次元上に配置する. この薄い紙を導出ペーパ (以下ペーパと省略) と呼ぶ. また呼び出す側と呼び出される側のペーパは奥行きをかえて配置する. それを線で結ぶことでペーパを木構造で表すことができ, ペーパがペーパを複数呼び出す導出の全体像が直観的に理解できると考える. 以下に 3 次元での視覚化を行う方針を示す. 図 3 を 3 次元で視覚化した例を図 5 に示す.

- 最初の導出のペーパを根 (root) として考える.
- 導出を呼び出す側を親, 呼び出される側を子とする.
- 親が導出を複数呼び出す場合, 子も複数になる.
- x 軸は導出が呼び出された順で決まり, y 軸の値は固定.
- 親と子は z 軸の値を変更して配置する.
- 親と子は線で結ぶことにする.

次に実際に 3 次元上に配置するための手順を以下に示す.

(1)　ペーパに名前を付ける.

- ペーパを出現順に仮名で 0,1,... とそれぞれ名付ける.

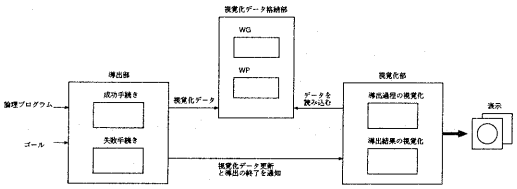

図 6　システム構成

- ペーパの子は親の名前をついでいき, 実際の名前とする.

（2） ペーパの名前から座標を決め, 配置する.

x 軸を横軸, y 軸を縦軸, z 軸を奥行きと考える.

- ペーパの仮名を x 座標とする.
- ペーパの実際の名前の長さを z 座標とする.

（3） ペーパの親と子を線で結ぶ.

- 親と子の座標を取得する.
- 親と子の間に線を描く.

4. システムの実装

本研究では, 3 節までで説明した手続きを実装した. 今回は実装を Java で行った. システムは以下の 3 つの部分から成る.

- 導出部
  - 成功手続きを行う
  - 失敗手続きを行う
- 視覚化データ格納部
  - WG_w を格納
  - WP_w を格納
- 視覚化部
  - 導出中の導出過程の視覚化機能
  - 導出後の導出過程の視覚化機能

システムは, まずユーザが導出を行う部分に対し, ゴールと論理プログラムを与える. 導出を行う部分を導出部とする. 導出部は論理プログラムをルールとし, ゴールに対し導出を行う.

導出を行うと, 視覚化データ格納部に視覚化データが格納される. 新たに視覚化データが格納されたとき, 導出が進んでいると考えられるので, 視覚化データが更新されたら, 視覚化データに基づき視覚化を行う. 視覚化を行う部分を視覚化部とする. 導出が終了すると, 先程までのウインドウの横に新たにウインドウが現れる. ウインドウには導出過程を表すペーパを 3 次元上に配置した図が表示される. 図 6 にシステムの構成を示す.

5. 例

作成したシステムで導出過程の視覚化を行った例を示す.

$P_3 = \{cloud \leftarrow rain \quad fine \leftarrow sunshine$
$wind \leftarrow typhoon \quad sunshine \leftarrow \sim dark$
$rain \leftarrow \neg snow, \sim fine \quad dark \leftarrow cloud, \sim moonlight$
$\neg snow \leftarrow \square \quad moonlight \leftarrow \sim rain \quad \}$

論理プログラム P_3 において $G_0 = \leftarrow cloud$ のときに, システムは図 7 のものを表示する. 図の左側のウインドウで, それぞれの導出を確認することができる. また右側のウインドウではそれぞれの導出を表したペーパが 3 次元上に配置されて, マウスを操作することにより視点変更ができる.

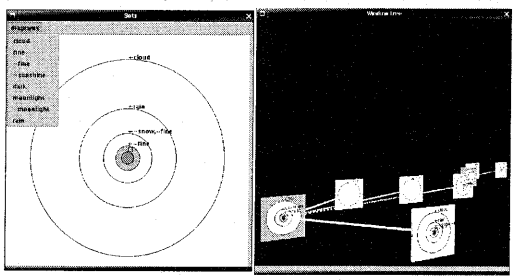

図 7　P_3 において $G_0 = \leftarrow cloud$ で導出過程の視覚化を行った例

6. おわりに

本論文では拡張論理プログラムにおける説明推論過程という情報を同心円や木の構造を利用して視覚化を行った. 説明推論過程をよりわかりやすく表現するように以下のことを考慮した.

- 必要な情報のみを表示：導出過程
- 全体と詳細を同時表示：3 次元での視覚化
- 抽象的データを画面に表示：紙を配置

さらに動的なデータに対し, 動的な視覚化を行った.

本論文ではシステムの実装のために, まず説明推論手続きを述べ, その説明推論過程を視覚化する手法を述べた. そして説明推論手続きを拡張した視覚化機能付き説明推論手続きを提案し, 手続きに基づき作成したシステムでは以下のことを行った.

- 導出の自動化
- 導出中の導出過程を動的に視覚化
- 導出後に導出過程を 3 次元で視覚化

そして実行例を示した. 今後の課題としてはリテラルに変数のある場合を扱えるようにすることと, システムの有効性を探り, システムを拡張していくことである.

文　献

[1] 山崎 進, “計算論理に基づく推論ソフトウェア論”, コロナ社, 2000.

[2] 小池 英樹 , “ビジュアライゼーション”, bit 別冊 ビジュアルインタフェイス-ポスト GUI を目指して-, 平川, 安村編, 第 2.1 章, pp.24-44, 共立出版,1996.

[3] 高田哲司, 小池英樹, “見えログ:情報視覚化とテキストマイニングを用いたログ情報ブラウザ”, 情報処理学会論文誌 Vol.41,No.12,pp.3265-3275, 2000.

[4] B. A. Myers, “Visual programming, programming by example and program visualization : A taxonomy.”, In Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'86), pp. 59–66. ACM Press, 1986.

[5] S.Yamasaki and M.Sasakura, “Towards distributed programming systems with visualizations based on nonmonotonic reasoning”, Proc. SSGRR 2001, CD-ROM form, 2001.

[6] M. Sasakura, “A visualization method for knowledge represented by general logic programs”, Fifth International Conference on Information Visualization(IV2001), pp.135–140, 2001.

[7] K.Eshghi and R.A.Kowalski, “Abduction Compared with Negation by Failure”, Proc. of 6th International Conference on Logic Programing. pp. 234-255, 1989.

[8] 吉田 聡, “論理プログラムにおける説明推論過程の視覚化システムの実装”, 岡山大学大学院自然科学研究科修士論文, 2003.