ICOT Technical Report: TR-0808

TR-0808

制約論理プログラムの帰納的一般化へ向けて

川村 正、古川 康一（慶応大）

October, 1992

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03)3456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

制約論理プログラムの帰納的一般化へ向けて
Towards Inductive Generalization in Constraint Logic Programs

川村 正
(財) 新世代コンピュータ技術開発機構
古川康一
慶應義塾大学 環境情報学部

1992 年 9 月 18 日

Abstract

本論文では制約論理プログラムの帰納的一般化法について述べる。従来、論理プログラムの枠組に基づいた機械学習の研究として、最小汎化や逆導出が提案されてきた。ここでは、これらの議論を拡張し、制約論理プログラムの最小汎化及び逆導出の一般的な枠組を提案する。また、制約の具体的例として、線形方程式 / 不等式からなる制約の場合についての計算法を述べる。

1 序論

近年、機械学習の研究のための枠組として帰納論理プログラミング (Inductive Logic Programming) が注目を浴びている。この研究の一般的な枠組は、背景知識 $\mathcal{K}$ が論理プログラムの形で与えられたとき、正の例を表すアトムの集合 $\mathcal{E}^+$ 及び負の例のを表すアトムの集合 $\mathcal{E}^-$ に対し、$\mathcal{K} \wedge \mathcal{H} \vdash \mathcal{E}^+$ 及び $\mathcal{K} \wedge \mathcal{H} \not\vdash \mathcal{E}^-$ を満足する論理プログラム $\mathcal{H}$ を求めるものである。

一方、論理プログラムの分野では、通常の論理プログラムの拡張として、制約論理プログラムの研究が盛んに行なわれている (cf. [5])。これは、さまざまな領域の制約充足法と論理プログラムを統合したものであり、具体的には算術代数 [1, 6] やブール代数 [2] などを制約の領域とする制約論理プログラムが実現されている。

帰納論理プログラミングの分野でも、最近、溝口と大和田が線形代数を領域とする制約論理プログラムの帰納的一般化を提案した [9]。また、この手法の空間推論への応用を提案した。

本論文では、制約論理プログラムの帰納的推論の方式の一般的枠組について考察する。この枠組は論理プログラムの最小汎化 [11] 及び逆導出 [10] の拡張である。また、具体的な制約の領域として、線形方程式 / 不等式を考える。

以下、第 2章では制約論理プログラムについて述べる。第 3章では制約論理プログラムの帰納的一般化について述べる。また、この章では、線形方程式 / 不等式からなる制約をとり上げる。第 4章では他の研究との関連について議論する。最後に、第 5章で結論を与える。

以下では、論理プログラムに関する基礎的な知識を仮定する [8]。A, B はアトムを、x, y, z, u, v, w は変数を、a, b, d, h, k は定数を、θ は代入を表す。また、式 F に代入 θ を適用して得られる式を $F\theta$ で表す。

2 . 制約論理プログラム

本節では、制約論理プログラムについて簡単に述べる [4]。

制約論理プログラムの枠組は多ソート一階論理で定義される。SORT$= \cup\{\mathrm{SORT}_i\}$ をソートの集合とする。n 引数関数記号の signature は $n+1$ 個のソートの列とする。また、n 引数述語記号の signature は n 個のソートの列とする。変数の signature はあるソートである。Σ を関数記号の集合、Π を述語記号の集合とする。Π, Σ 上のアトムを (Π, Σ)- アトムと呼ぶ。(Π, Σ)- アトムの集合を (Π, Σ)- 制約と呼ぶ。直観的には、(Π, Σ)- 制約は (Π, Σ)- アトムの連言を表す。

定義 2.1 制約論理プログラム

Σ を関数記号の集合、Π を述語記号の集合とする。Π_B, Π_C を $\Pi = \Pi_B \cup \Pi_C, \Pi_B \cap \Pi_C = \emptyset$ を満たす述語記号の集合とする。また、$= \in \Pi$ とする。このとき次のいずれかの形をした式を (Π, Σ)- 制約規則と呼ぶ。

$A \leftarrow c$ []
$A \leftarrow c$ [] $B_1, B_2, \ldots, B_n$.
ここで $A, B_1, B_2, \ldots, B_n$ は (Π_B, Σ)- アトム、c は (Π_C, Σ)- 制約である。(c は空集合でも良い。) A を C のヘッド、$B_1, B_2, \ldots, B_n$ を C のボディ、c を C の制約部と呼ぶ。

(Π, Σ)- 制約規則の集合を (Π, Σ)- プログラムと呼ぶ。

また、次のいずれかの形をした式を (Π, Σ)- ゴールと呼ぶ。

$\leftarrow c$ []
$\leftarrow c$ [] $B_1, B_2, \ldots, B_n$
ここで $A, B_1, B_2, \ldots, B_n$ は (Π_B, Σ)- アトム、c は (Π_C, Σ)- 制約である。(c は空集合もしくは無限集合でも良い。)

Π, Σ 上の structure を [4] のように定義し、$\mathcal{R}(\Pi, \Sigma)$ と表記する。ただし、Π は signature を持たない等号記号 $=$ を含んでいるとする。Π, Σ 上の表現に対する $\mathcal{R}(\Pi, \Sigma)$-valuation は、その表現中の異なる各変数から $D\mathcal{R}_s$ へのマッピングである。ここで s は該当する変数のソートである。

以下では、$\mathcal{R}(\Pi, \Sigma)$ プログラム、$\mathcal{R}(\Pi, \Sigma)$- 制約規則、$\mathcal{R}(\Pi_B, \Sigma)$- アトム、$\mathcal{R}(\Pi_C, \Sigma)$- 制約を単に (制約論理) プログラム、制約規則、アトム、制約と書く。また、$\mathcal{R}(\Pi, \Sigma)$ を $\mathcal{R}$ と略記する。

例 2.1 $\mathcal{R}$ をある関数記号の有限集合 Σ 上のエルブラン空間とする。また、$\Pi = \{=\}$ とする。このとき、制約論理プログラムは Prolog プログラムに対応する。例えば、$\Pi_B = \{\text{less-than}\}$、$\Sigma = \{s\}$ のとき、制約論理プログラムは以下のように書かれる。

less-than$(x, y) \leftarrow \{x = 0\}$ []
less-than$(x, y) \leftarrow \{x = s(w), y = s(z)\}$ [] less-than(w, z)

例 2.2 算術を領域とする制約論理プログラムは以下のように定義できる。

$\Sigma = \{+, *, 0, 1, 2, \ldots\}$

$+, *$ の signature	(s_1, s_1, s_1)
$0, 1, 2, \ldots$ の signature	(s_1)

$\Pi = \{=, <, >, \leq, \geq\}$

$<, >, \leq, \geq$ の signature	(s_1, s_1)

ここで、Π_p が以下のように定義されたとする。

$\Pi_p = \{\text{ohm,voltage,resistance}\}$

ohm の signature	(s_2, s_2, s_1)
voltage,resistance の signature	(s_2, s_1)

このとき、制約規則は以下のように書かれる。

ohm$(x, y, i) \leftarrow \{v = r * i\}$ [] voltage(x, v), registance(y, r).

c を制約とする。$\forall c' \in c \;\; \mathcal{R} \models c'\theta$ が成立するとき $\mathcal{R} \models c\theta$ と書く。

定義 2.2 制約の可解性

c を制約、θ を $\mathcal{R}$-valuation とする。$\mathcal{R} \models c\theta$ が常に成立するとき、c は $\mathcal{R}$- 可解であるという。このとき、θ を c の $\mathcal{R}$- 解、または単に解と呼ぶ。

定義 2.3 制約論理プログラムの導出

P をプログラム、G_0 を次のようなゴールとする。

$\leftarrow c_0$ [] $A_1, A_2, \ldots, A_n (n \geq 1)$
ここで、c_0 が $\mathcal{R}$- 可解で P が次の形の制約規則を含むとする。

$B_0 \leftarrow c$ [] $B_1, B_2, \ldots, B_m (m \geq 0)$
次の形のゴールを G_1 とする。

$\leftarrow c_1$ [] $A_1, \ldots, A_{i-1}, B_1, \ldots, B_m, A_{i+1}, \ldots, A_n$
ただし、$c_1 = \{c, c_0, A_i = B_0\}$ かつ c_1 は $\mathcal{R}$- 可解である。このとき、G_1 は G_0 から導出可能 (derivable) であるという。

定義 2.4 導出列と解制約
ゴールの列 $G_0, G_1, \ldots, G_n$ で、G_{i+1} が G_i から導出可能であるものを導出列 (derivation sequence) と呼ぶ。また、$G_n = c\,[]$ のとき、この列を成功列と呼ぶ。このとき、c を解制約 (answer constraint) と呼ぶ。

論理プログラムの場合と同様、制約論理プログラムに対してもモデル論的意味論、不動点意味論、手続き的意味論が定義でき、これらの意味論の関係が示されている [4]。

3 制約論理プログラムの帰納的一般化

本節では、論理プログラムの帰納的一般化で用いられる最小汎化 (lgg) [11] 及び逆導出 [10] を制約論理プログラムに適用できるように拡張する。

C を $A \leftarrow c\,[]\,B_1, B_2, \ldots, B_n$ の形をした制約規則とする。このとき literals$(C) = \{A, \neg B_1, \neg B_2, \ldots, \neg B_n\}$, constraint$(C)$=$c$ と定義する。

制約規則 C は、literals(C) 中のアトムの引数がすべて異なる変数であるとき、標準形であるという。Π_p が等号記号 = を含むとき、任意の制約規則は標準形に書き換えることができる。以下では、簡単のため、すべての制約規則は標準形であるとする。

F を式もしくは式の集合とする。F に現れる変数の集合を vars(F) と書く。

3.1 制約規則の最小汎化

本節では、確定節の最小汎化 [11] を制約規則に拡張し、制約規則の最小汎化の一般的な定義を与える。まず、アトムの集合の最小汎化の一般的な定義を与える。

定義 3.1 一般化順序
S をすべての制約の集合とする。S 上に半順序 $\preceq$ が定義でき、S は $\preceq$ の下で完備束をなすとき、$\preceq$ を制約の一般化順序と呼ぶ。S の任意の要素 c_1, c_2 に対し、$c_1 \prec c_2$ が成り立つとき、c_1 は c_2 より一般的であるという。

定義 3.2 制約の最小汎化
S をすべての制約の集合、$\preceq$ を S 上の一般化順序とする。L を S の部分集合とする。L の greatest lower bound c を L の最小汎化 (lgg) と呼び、$c = \mathrm{lgg}_{\preceq}(L)$ と書く。即ち、c は以下を満足する。

(1) $\forall \bar{c} \in L\ c \preceq \bar{c}$

(2) $\forall c'$ s.t. $\forall \bar{c} \in L\ c' \preceq \bar{c},\ c' \preceq c$

S は完備束であるから、定義より任意の制約の集合の lgg は必ず存在することに注目されたい。
最小汎化を計算できるためには、以下の条件が必要である。

(a) 任意の制約 c, c' に対し、$c \preceq c'$ が証明可能である。

(b) 任意の制約の集合に対し、その $\preceq$ に関する greatest lower bound を求めるアルゴリズムが存在する。

次に、[11] に従い、確定節の最小汎化を定義する。

定義 3.3 θ-subsumption
C, D を節とする。代入 θ が存在し、$C\theta \subseteq D$ が成立するとき、C は D を θ subsumes するといい、$C \preceq_\theta D$ と書く。

定義 3.4 確定節の最小汎化
S を確定節の集合とする。以下を満足する確定節 C を S の最小汎化と呼び、$C = \mathrm{lgg}_\theta(S)$ と書く。

(1) $\forall D \in S\ C \preceq_\theta D$

(2) $\forall C'$ s.t. $\forall D \in S\ C' \preceq_\theta D$ に対し $C' \preceq_\theta C$

定義 3.4は、定義 3.2で一般化順序として θ-subsumption を採用した場合であることに注目されたい。

確定節の lgg は一意には決まらない。しかし確定節は θ-subsumption に関して同値類に分類でき、任意の確定節をその節が属する同値類の代表元に変換するアルゴリズムが存在する [11]。こうして lgg を計算するアルゴリズムは [11] に示されている。

例 3.1 p(f(x),g(b),h(c)) と p(u,g(v),h(c)) の lgg は p(x,g(y),h(c)) である。

次に、制約規則の共通汎化を定義する。

定義 3.5 制約規則の最小汎化
$S = \{C_1, C_2, \ldots, C_n\}$ を制約規則の集合とする。ここで、制約に対して一般化順序 $\preceq$ が定義されており、任意の制約 c_1, c_2 に対して $c_1 \preceq_\theta c_2$ が成り立つとき $c_1 \preceq c_2$ が成り立つとする。このとき、以下を満たす制約規則 C を S の最小汎化と呼び、$C = \mathrm{lgg}_{(\theta,\preceq)}(S)$ と書く。

(1) $\mathrm{literals}(C) = \mathrm{lgg}_\theta(S_{lit})$
ただし、$S_{lit} = \{\mathrm{literals}(C_1), \mathrm{literals}(C_2), \ldots, \mathrm{literals}(C_n)\}$, $(\mathrm{literals}(C))\theta_i \subseteq \mathrm{literals}(C_i)$ $(i = 1, 2, \ldots, n)$

(2) $\mathrm{constraint}(C) = \mathrm{lgg}_\preceq(S_c)$
ただし、$S_c = \{\mathrm{constraint}(C_1\theta_1^{-1}), \mathrm{constraint}(C_2\theta_2^{-1}), \ldots, \mathrm{constraint}(C_n\theta_n^{-1})\}$ [1]

ここでは、制約の最小汎化について一般的な枠組のみを示した。しかしながら、任意の制約の集合の最小汎化が存在するためには、制約の集合が一般化順序に関して完備束をなす、という条件が必要であるが、多くの領域ではこの条件は強過ぎて一般化順序が定められないかも知れない。例えば、論理的含意は直観的には一般化順序として適当であると思われるが、実数を領域とする代数においては、一般には含意関係は同値類を考えても完備束にはならない。

一般化順序を定めるのが難しい場合には、最小汎化に関して以下のように条件を緩めるのが妥当であると思われる。

(a) 一般化順序は半順序であり、全制約の集合はこの半順序に関して束をなすが、完備束をなさなくとも良い。

(b) 定義 3.2から条件 (2) を除く。即ち、最小性を要求しない。

上のようにして求まる汎化を共通汎化と呼ぶ。共通汎化は一意には定まらないので、どの共通汎化をとるかは議論が残る。しかし、この定義を用いると含意関係による一般化が多くの領域で採用できると考えられる。

線形方程式 / 不等式からなる制約については、含意関係を一般化順序として採用できる。これについては、3.3節で述べる。

3.2 制約論理プログラムの逆導出

論理プログラムの逆導出では、truncation,absorption,intra-construction の 3 つの操作が定義されている [10]。本節では、この 3 つの操作を制約論理プログラムに適用できるように拡張する。制約部の計算法については 3.3で述べる。

(a) truncation

C_1, C_2 をボディが空である制約規則とする。C_1 と C_2 の truncation は $\mathrm{lgg}(C_1, C_2)$ であるとする。

(b) absorption

制約規則 C_1, C_2 から C を導く導出があったと想定する。absorption は、C と C_1 から逆に C_2 を推測する操作である。即ち、

$$\mathrm{literals}(C) = (\mathrm{literals}(C_1) - \{l_1\})\theta_1 \cup (\mathrm{literals}(C_2) - \{l_2\})\theta_2$$
$$(\mathrm{constraint}(C_1))\theta_1 \cup (\mathrm{constraint}(C_2))\theta_2 \vdash \mathrm{constraint}(C)$$

である。ただし、l_1 は $\mathrm{literals}(C_1)$ 中の正リテラル、l_2 は $\mathrm{literals}(C_2)$ 中の負リテラルで、$l_2\theta_2 = \neg l_1\theta_1$ とする。ここで、$\mathrm{literals}(C_1) = \{l_1\}$ と仮定すると

$$\mathrm{literals}(C_2) = (\mathrm{literals}(C) \cup \{\neg l_1\}\theta_1)\theta_2^{-1}$$
$$(\mathrm{constraint}(C_1))\theta_1 \cup (\mathrm{constraint}(C_2))\theta_2 \vdash \mathrm{constraint}(C)$$

を得る。

[1] 一般に代入 θ に対し θ^{-1} が定義できないことがあるが、制約規則が標準形であるという仮定により、θ_i^{-1} は常に定義できる。

例 3.2 制約規則 C,C_1 が以下のように与えられたとする。

$$C: \text{less-than}(u,v) \leftarrow \{v = s(s(u))\} \; []$$
$$C_1: \text{less-than}(u_1,v_1) \leftarrow \{v_1 = s(u_1)\} \; []$$

$\theta_1 = \{u_1/u', v_1/v'\}$, $\theta_2 = \{x/u, y/v, x_1/u', y_1/v'\}$ として absorption を適用すると、以下が得られる。

$$C_2: \text{less-than}(x,y) \leftarrow c \; [] \; \text{less-than}(x_1,y_1)$$
$$\{v' = s(u')\} \cup c\theta_2 \vdash \{v = s(s(u))\}$$

上の制約式より c を求めると、以下が得られる。

$$C_2: \text{less-than}(x,y) \leftarrow \{x = x_1, y = s(y_1)\} \; [] \; \text{less-than}(x_1,y_1)$$

例 3.3 制約規則 C,C_1 が以下のように与えられたとする。

$$C: \text{ohm}(a,z,i) \leftarrow i = 3 \; []$$
$$C_1: \text{resistance}(r,r_v) \leftarrow c_1 \; []$$

$\theta_1 = \{r/z\}$, $\theta_2 = \{r'/z\}$ として C,C_1 に absorption を適用すると

$$C_2: \text{ohm}(a,r',i) \leftarrow c_2 \; [] \; \text{resistance}(r',r_v).$$

ただし

$$c_1\theta_1 \cup c_2\theta_2 \vdash i = 3$$

(c) intra-construction

制約規則 C と C_1 から D_1 を、C と C_2 から D_2 導く導出があったと想定する。intra-construction は D_1 と D_2 から C,C_1,C_2 を推測する操作である。即ち、

$$\text{literals}(D_i) = (\text{literals}(C) - \{l\})\theta_i \cup (\text{literals}(C_i) - \{l_i\})\theta'_i$$
$$(\text{constraint}(C))\theta_i \cup (\text{constraint}(C_i))\theta'_i \vdash \text{constraint}(D_i)$$

である。ここで、l は literals(C) 中の正リテラル、l_i は literals(C_i) 中の負リテラルで、$\neg l\theta_i = l_i\theta'_i$ とする。ここで、literals(C_i) = $\{l_i\}$ と仮定すると

$$\text{literals}(D_i) = (\text{literals}(C) - \{l\})\theta_i$$

を得る。これより以下の関係を得る。

$$\text{literals}(C) = (\text{literals}(D_1))\theta_1^{-1} \cup \{l\} = (\text{literals}(D_2))\theta_2^{-1} \cup \{l\} = \text{literals}(D) \cup \{l\}$$

また、$\theta'_1 = \theta'_2 = \emptyset$ と仮定すると以下の関係を得る。

$$\text{literals}(C_i) = \neg l\theta_i$$
$$(\text{constraint}(C))\theta_i \cup \text{constraint}(C_i) \vdash \text{constraint}(D_i)$$

また、簡単のため、vars(literals(C)) = vars(constraint(C)) とする。

例 3.4 制約規則 D_1,D_2 が以下のように与えられたとする。

$$D_1: \min(u_1,v_1) \leftarrow \{u_1 = u_2, v_1 = [s(u_2)|v_2]\} \; [] \; \min(u_2,v_2).$$
$$D_2: \min(u_1,v_1) \leftarrow \{u_1 = u_2, v_1 = [s(s(u_2))|v_2]\} \; [] \; \min(u_2,v_2).$$

ここで $\theta_1 = \theta_2 = \{x_1/u_1, y_1/v_1, x_2/u_2, y_2/v_2\}$ とすると以下を得る。

$$C_1: \text{p}(u_2,z) \leftarrow c_1 \; []$$
$$C_2: \text{p}(u_2,z) \leftarrow c_2 \; []$$
$$C: \min(x_1,y_1) \leftarrow c \; [] \; \min(x_2,y_2), \text{p}(x_2,z).$$

ただし

$c\theta_1 \cup c_1 \vdash \{u_1 = u_2, v_1 = [s(u_2)|v_2]\}$
$c\theta_2 \cup c_2 \vdash \{u_1 = u_2, v_1 = [s(s(u_2))|v_2]\}$

上式より以下が求まる。

C_1 : $\mathrm{p}(u_2, z) \leftarrow \{z = s(u_2)\}$ []
C_2 : $\mathrm{p}(u_2, z) \leftarrow \{z = s(s(u_2))\}$ []
C : $\min(x_1, y_1) \leftarrow \{x_1 = x_2, y_1 = [z|y_2]\}$ [] $\min(x_2, y_2), \mathrm{p}(x_2, z)$.

例 3.5 制約規則 D_1, D_2 が以下のように与えられたとする。

D_1 : $\mathrm{ohm}(a, x, i) \leftarrow \{i = 3\}$ []
D_2 : $\mathrm{ohm}(a, y, i) \leftarrow \{i = 4\}$ []

$\theta_1 = \{\mathrm{r/x}\}, \theta_2 = \{\mathrm{r/y}\}$ として intra-construction を適用すると、以下が得られる。

C : $\mathrm{ohm}(a, r, i) \leftarrow c$ [] $\mathrm{resistance}(r, r_v)$.
C_1 : $\mathrm{resistance}(x, r_v) \leftarrow c_1$ []
C_2 : $\mathrm{resistance}(y, r_v) \leftarrow c_2$ []

ただし

$c\theta_1 \cup c_1 \vdash \{i = 3\}$
$c\theta_2 \cup c_2 \vdash \{i = 4\}$

例 3.5では、intra-construction の際に θ_i にも D_i にも現れない変数 r を導入している。論理プログラムの逆導出では通常このような変数の導入は無意味であったが、制約論理プログラムの場合はこのような変数を導入して新しいパラメータとして制約部に加えなければ一般化に失敗することがある。

3.3 線形方程式 / 不等式からなる制約

本節では、線形方程式 / 不等式からなる制約について、共通汎化及び逆導出で行なう計算法について述べる。

3.3.1 準備

本節では準備として、基本的な概念を与える。

制約を線形方程式または不等式の集合であるとする。即ち、$\Sigma = \{+, *\}, \Pi = \{=, >, \geq\}$ である。慣例に従い、$a > b, a \geq b$ を $b < a, b \leq a$ と書くことがある。また、$a * b$ を ab と書くことがある。

線形方程式 / 不等式の集合 S に対し、S 中のすべての線形方程式の集合を $\mathrm{eqs}(S)$、すべての不等式集合を $\mathrm{ineqs}(S)$ と書く。線形方程式のみからなる集合 S_e の系数行列を $\mathrm{mat}(S_e)$ と書く。S_e が可解で $\mathrm{rank}(\mathrm{mat}(S_e)) < |S_e|$ のとき、S_e 中の方程式のうちいくつかを消去し、$\mathrm{rank}(\mathrm{mat}(S_e')) = |S_e'|$ を満たす線形方程式の集合 S_e' を得ることができる。このことから、以下では線形方程式 / 不等式の集合 S に対し、$\mathrm{rank}(\mathrm{mat}(\mathrm{eqs}(S))) = |\mathrm{eqs}(S)|$ が必ず成り立っていると仮定できる。

以下では、簡単のため、線形方程式 / 不等式を関数記号 f, g 及び変数のベクタ $\overline{x}$ を使って $f(\overline{x}) = d, g(\overline{x}) \geq d'$ などと表す。

3.3.2 線形方程式からなる制約の最小汎化

一般化順序として、含意関係を採用する。即ち、線形方程式 / 不等式の集合 S_1, S_2 に対し、$S_2 \rightarrow S_1$ が成り立つとき、S_1 は S_2 より一般的であるとする。線形方程式 / 不等式からなる制約の場合には、以下に述べるように、最小汎化が存在し、これを求めることができる。

S_1, S_2 を線形方程式 / 不等式の集合とする。$\mathrm{vars}(S_1 \cup S_2) = n$ とすると、S_1, S_2 はそれぞれ n 次元空間中の凸集合を表す。n 次元空間の任意の部分集合 S に対し、S を包含する最小の凸集合が必ず存在する。これを S の凸包と呼ぶ。従って、$S_1 \cup S_2$ の凸包を S_1 と S_2 の最小汎化とすれば良い。

例 3.6 $S_1 = \{x = 1, y = 0\}, S_2 = \{x = 2, y = 1\}$ とすると、$\mathrm{lgg}(S_1, S_2) = \{x - y = 1, 1 \le x \le 2, 0 \le y \le 1\}$ である。また、$S_1 = \{x - y + z = 0, x + y + z = 2\}, S_2 = \{x - y + z = 1, x + y + z = 3\}$ とすると、$\mathrm{lgg}(S_1, S_2) = \{x - y + z - 1, 0 \le x - y + z \le 0, 2 \le x + y + z \le 3\}$ である。

凸包の計算法に関しては、2次元に関しては [3]、多次元に関しては [12] など、多くの研究がなされている。

3.3.3 逆導出における線形方程式 / 不等式からなる制約の計算

3.2で述べたように、制約論理プログラムの逆導出では、制約部に関して以下の計算を行なう必要がある。

(1) 制約 c_1, c_2 が与えられた時、以下を満たす制約 c を求める。(truncation)

$c_1 \to c,\ c_2 \to c$

(2) 制約 c, c_1 が与えられた時、以下を満たす制約 c_2 を求める。(absorption)

$c_1 \cup c_2 \vdash c$

(3) 制約 c_1', c_2' が与えられた時、以下を満たす制約 c, c_1, c_2 を求める。(intra-construction)

$c \cup c_1 \vdash c_1',\ c \cup c_2 \vdash c_2'$

(1) に関しては最小汎化を求めれば良いので、本節では (2)(3) の場合について述べる。

(a) absorption

線形方程式 / 不等式の集合 S, S_1 が与えられたとき、$S_1 \cup S_2 \vdash S$ 満たす線形方程式 / 不等式の集合 S_2 を求める方法について述べる。ここで、$S \cup S_1$ は充足可能であると仮定する。

$\mathrm{eqs}(S_1) = \{f_1(x_1, \ldots, x_n) = d_1, \ldots, f_m(x_1, \ldots, x_n) = d_m\}$ とする。また、e を線形方程式 $g(x_1, \ldots, x_n) = d$ とする。このとき、$ag(x_1, \ldots, x_n) + a_1 f_1(x_1, \ldots, x_n) + \cdots + a_m g_m(x_1, \ldots, x_n) = ad + a_1 d_1 + \cdots a_m d_m \in S_2$ であるならば $\mathrm{eqs}(S_1) \cup S_2 \vdash e$ である。($a, a_1, \ldots, a_m$ は任意の実数。) 線形不等式についても同様である。このことから、S_2 を次のように求めることができる。

手続き 3.1　absorption(S_1, S_2)
入力：S, S_1：線形方程式 / 不等式の集合
出力：S：線形方程式 / 不等式の集合

$\overline{x}$ を $\mathrm{vars}(S) \cup \mathrm{vars}(S_1)$ 中のすべての変数のベクタとする。ここで、S_1 に関して次が成り立つとする。

$\mathrm{eqs}(S_1) = \{f_1(\overline{x}) = d_1, \ldots, f_l(\overline{x}) = d_l\}$
$\mathrm{ineqs}(S_1) = \{f_{l+1}(\overline{x}) \ge d_{l+1}, \ldots, f_m(\overline{x}) \ge d_m\}$

このとき、S_2 は次のように得られる。

$$\begin{aligned}\mathrm{eqs}(S_2) = \{\ & ag_e(\overline{x}) + a_1 f_1(\overline{x}) + \cdots + a_l f_l(\overline{x}) = ad_e + a_1 d_1 + \cdots + a_l d_l \mid \\ & \forall g_e(\overline{x}) = d_e \in \mathrm{eqs}(S), a, a_1, \ldots, a_l \text{ は任意の実数、ただし } a \ne 0\} \\ \mathrm{ineqs}(S_2) = \{\ & bg_{in}(\overline{x}) + b_1 f_1(\overline{x}) + \cdots + b_l f_l(\overline{x}) + b_{l+1} f_{l+1}(\overline{x}) + \cdots + b_m f_m(\overline{x}) \ge bd_{in} + b_1 d_1 + \cdots + b_l d_l + \\ & b_{l+1} d_{l+1} + \cdots + b_m d_m \mid \forall g_{in}(\overline{x}) - d_{in} \in \mathrm{ineqs}(S), b \text{ は任意の正の実数、} b_1, \ldots, b_l \text{ は任意の} \\ & \text{実数、} b_{l+1}, \ldots, b_m \text{ は } 0 \text{ もしくは任意の正の実数}\}\end{aligned}$$

例 3.7 $S = \{y = 2, z \ge 2\}, S_1 = \{x = 1\}$ とする。このとき、a_1, b_1, a_2 を任意の実数、b_2 を任意の正の実数とすると $S_2 = \{a_1 x + b_1 y = a_1 + 2b_1, a_2 x + b_2 z \ge a_2 + 2b_2\}$ は $S_1 \cup S_2 \vdash S$ を満たす。

[9] の absorption では、制約部を以下のように求めている。制約規則 C, C_1 が与えられたとする。absorption の結果得られる制約規則を C_2 とすると、$\mathrm{constraint}(C_2) = \mathrm{constraint}(C) \cup \mathrm{constraint}(C_1)$ である。この方法では、例 3.7の場合、$S_2 = \{x = 1, y = 2, z \ge 2\}$ が求まる。本論文の方法では得られる結果はより逆導出の解として適当であると思われる。

(b) intra-construction

線形方程式 / 不等式の集合 T_1, T_2 が与えられた時、$S \cup S_1 \vdash T_1, S \cup S_2 \vdash T_2$ を満たす S, S_1, S_2 を求める手続きを示す。ここで、3.2の仮定より、$\mathrm{vars}(S)$ は既知であるとする。

手続き 3.2 intra-construction-on-linear-equations(T_1, T_2)
入力：T_1, T_2：線形方程式 / 不等式の集合
出力：S, S_1, S_2：線形方程式 / 不等式の集合

(1) S を任意の線形方程式 / 不等式の集合とする。

(2) $S_1 = \text{absorption}(T_1, S)$.

(3) $S_2 = \text{absorption}(T_2, S)$.

例 3.8 $T_1 = \{x = 2\}, T_2 = \{x = 4\}$ とする。vars$(S) = \{x, y\}$ としたとき、$S \cup S_i \vdash T_i$ を満たす S, S_1, S_2 は以下のように求められる。

(1) $S = \{ax + by = d\}$ とする。

(2) $S_1 = \text{absorption}(T_1, S) = \{(k_1 + k_2 a)x + k_2 by = 2k_1 + k_2 d\}$.

(3) $S_2 = \text{absorption}(T_2, S) = \{(h_1 + h_2 a)x + h_2 by = 4h_1 + h_2 d\}$.

簡単のため例えば $a = 1, b = -2, d = 0, k_1 = 1, k_2 = -1, h_1 = 1, h_2 = -1$ とすれば、$S = \{x = 2y\}, S_1 = \{y = 1\}, S_2 = \{y = 2\}$ が得られる。

4 他の研究との比較

制約論理プログラムの特徴のひとつは、異なる種類のパラメータの関係を明示的に表現できることである。異なる種類のパラメータを含んだ問題を扱う学習システムとしては、Bacon.4 [7] があげられる。Bacon.4 では、非数値パラメータと数値パラメータとの関係を推論する際に、固有特性の導入というヒューリスティクスを用いている。固有特性の導入は以下のように行なわれる。

「x を非数値を取る変数、y を数値を取る変数とする。x の値が変化する時 y の値も変化するならば、次のような固有特性 z を仮定する。z は y と同じ値を取り、その値は記述されている条件すべてと関係付けられる。」

以下に例を挙げる。次の例は文献 [7] に現れたものである。

例 4.1 次のような回路で、電源及び電線を取り換えて電流値を測定したとする。このとき、電流値の変化に関する法則性を学習する。

今、観測結果として、以下の結果が得られたとする。

	電源	電線	電流
(1)	e	r_1	1
(2)	e	r_2	2

ここで、a は電源の、r_1, r_2 は電線の名前とする。

観測結果から、電線を取り換えると電流値が変化していることがわかる。そこで、Bacon.4 は固有特性 z を導入する。z は電流値と同じ値を取り、他のすべてのパラメータに即ち電源、電線、電流値に異存するものと仮定する。さらに例を与えることによって、パラメータ z は電線にのみ依存することが推論できる。詳しくは [7] を参照されたい。

固有特性の導入は、非数値パラメータと数値パラメータを関係付けるためのヒューリスティックと考えられる。制約論理プログラムを用いると、非数値パラメータと数値パラメータの関係を明示的に表現できる。これらのパラメータの関係を表す新述語を intra-construction によって生成すれば、固有特性の導入に相当する推論を実現できる。

例 4.2　例 4.1の問題を制約論理プログラムで表す。観測結果は以下の制約規則で表される。

$$D_1 : \mathrm{ohm}(e, r_1, z) \leftarrow z = 1 \,[]$$
$$D_2 : \mathrm{ohm}(e, r_2, z) \leftarrow z = 2 \,[]$$

ここで、D_1, D_2 に $\theta_1 = \{y/r_1\}, \theta_2 = \{y/r_2\}$ として intra-construction を適用し、以下を得る。

$$C \;: \mathrm{ohm}(e, y, z) \leftarrow c \,[]\, \mathrm{resistance}(y, w).$$
$$C_1 : \mathrm{resistance}(r_1, y) \leftarrow c_1 \;[]$$
$$C_2 : \mathrm{resistance}(r_2, y) \leftarrow c_2 \;[]$$

直観的には、述語 resistance は電線 (第 1 引数) とその電流容量 (第 2 引数) の関係を表す。ここで、制約 c, c_1, c_2 の間に以下の関係が成り立つ。

$$c\theta_1 \cup c_1 \vdash \{z = 1\} \tag{1}$$
$$c\theta_2 \cup c_2 \vdash \{z = 2\} \tag{2}$$

ただし、$\mathrm{vars}(c) = \{y, z\}$ である。

(1),(2) を満たす制約 c, c_1, c_2 を求めると、以下が得られる。

$$\begin{aligned} c &= \{ay + bz = d\} \\ c_1 &= \{ak_1 y + (bk_1 + k_2)z = k_1 d + k_2\} \\ c_2 &= \{ah_1 y + (bh_1 + h_2)z = h_1 d + 2h_2\} \end{aligned}$$

簡単のため $a = 1, b = -1, k_1 = k_2 = 1, d = 0$ とすると、$c = \{y = z\}, c_1 = \{y = 1\}, c_2 = \{y = 2\}$ が得られる。

Bacon.4 では、上記以外に、理論項を導入するヒューリスティックも用いている。これは 2 つ数値パラメータの値の増減の傾向に関連性がある推論できるとき、これらの間に比例関係を仮定し、その比例定数として理論項を導入するものである。このような推論は、われわれの枠組では、制約部の計算で行なわれる。即ち、制約部に現れる式を線形方程式に限定することによってパラメータ間の比例関係を仮定していることになる。

Bacon.4 は、理論項を変数として導入することにより、非線形方程式も取り扱うことができる。例 4.1でさらに電源を交換して得られた測定データを与えると、Bacon.4 はオームの法則 $V = R * I$ を学習する。我々の方法は現在線形方程式しか扱えないのでこのような学習は行なえないが、制約部の計算法を拡張して非線形方程式を扱えるようにすれば、Bacon.4 と同等の機能を期待できる。

溝口と大和田は [9] で線形方程式 / 不等式を領域とする制約論理プログラムの最小汎化と absorption を提案した。また、これらの技法が空間推論に利用できることを示した。本論文で示した最小汎化の枠組はより一般的なものである。また、逆導出に関して、intra-construction も定義した。

5　結論

制約論理プログラムの帰納的一般化法について述べた。帰納論理プログラムでの議論を拡張し、制約論理プログラムの最小汎化及び逆導出の一般的な枠組を提案した。制約の具体的例として、線形方程式 / 不等式からなる制約の場合についての計算法を述べた。

謝辞

御議論頂いた (財) 新世代コンピュータ技術開発機構の佐藤洋祐博士、有馬淳氏に感謝致します。また、本研究を行なう機会を与えて下さった淵一博博士に感謝致します。

参考文献

[1]　Colmerauer, A., "Introduction of PrologIII", ESPRIT'87 Achievements and Impact, Proc. of the 4th Annual ESPRIT Conference, Brussels, North Holland, 1987.

[2] Dincbas, M., P. Van Hentenryck, H. Simonis, A. Aggoun, T. Graf and F. Berthier, "The Constraint Logic Programming Language CHIP", Proc. of the International Conference on Fifth Generation Computer Systems 1988, pp.693 – 702, Tokyo, 1988.

[3] Graham, R. L., "An Efficient Algorithm for Determining the Convex Hull of a Finite Planar Set", Information Processing Letters, Vol.1, pp.132 – 133, 1972.

[4] Jaffar, J. and J.- L. Lassez, "Constraint Logic Programming", Technical Report, Department of Computer Science, Monash University, 1986.

[5] Jaffar, J. and J.- L. Lassez, "Constraint Logic Programming", Proc. of the 14th ACM Symposium on Principles of Programming Languages, pp.111 – 119, 1987.

[6] Jaffar, J. and S. Michaylov, "Methodology and Implementation of A Constraint Logic Programming System", Proc. of the 4th International Conference on Logic Programming, Melbourne, 1987.

[7] Langley, P., "Rediscovering Chemistry With the BACON System", Machine Learning, R. S. Michalski, J. G. Carbonell and T.M. Mitchell (eds.), Morgan Kaufmann, pp.307 – 330, 1983.

[8] Lloyd, J. W., "Foundations of Logic Programming", Springer-Verlag, 2nd Edition, Berlin, Heidelberg, New York, 1987.

[9] Mizoguchi F., and H. Ohwada, "Constraint-directed Generalization for Learning Spatial Relations", Proc. of International Workshop on Inductive Logic Programming, Tokyo, 1992.

[10] Muggleton, S. and W. Buntine, "Machine Invention of First-Order Predicates by Inverting Resolution", Prof. of 5th International Conference on Machine Learning, pp.339 – 352, 1988.

[11] Plotkin, G.D., "A note on inductive generalization", in Machine Intelligence 5, eds. B. Meltzer and D. Michie, Elsevier North-Holland, pp.153–163, New York, 1970.

[12] Toussaint, G. T., S. G. Akl and L. P. Devroye, "Efficient Convex Hull Algorithms for Points in Two and More Dimensions", Technical Report SOCS-78.5, McGill University, 1978.