1Q－8

SYNAPS：相関にもとづく論理プログラムの自動合成システム

斉藤嘉幸，中村克彦

東京電機大学

１．　はじめに

与えられた入出力例から，この関係を満足するシステムを推定し合成するような帰納推論の問題は，学習理論の中でも最も重要な課題の一つである．

われわれは，与えられた入出力例の相関[1]を解析することにより論理プログラムを自動合成するシステムSYNAPS(SYNthesis by Analyzing Positive Samples)を構成している．相関は入出力例からこれに含まれる再帰的構造をleast general generalization(LGG)[2]を用いて解析する方法であり，あるアトムの内部に他のアトムの構造が含まれることを検出する．この報告では相関の結果から逆融合のための帰納推論オペレータ[3]を拡張したオペレータを用いて，入出力例を満足する論理プログラムが効率よく合成できることを示す．

２．　準備

本論文で扱う用語は文献[4]に準じている．アトムp($t_1,\cdots,t_n$)のすべての引き数($t_1,\cdots,t_n$)をS，Tなどで表わす．たとえば，p($s_1,\cdots,s_n$)をp(S)と記す．

２．１　節の分解

アトムの構造を分析するためにアトムの分解を行なう．この手続きは以下で定義される．

1. 単位節のflattening[5]を行ない節を作る．
2. 節の本体を引き数ごとに２つに分解する．

[例１]　rev([a,b,c],[c,b,a])

1. rev(x,y)=(x=[a|x1],x1=[b|x2],x2=[c],
　　y=[c|y1],y1=[b|y2],y2=[a])
2. 第１引き数はx1，第２引き数はy2で分解すると，
　　rev([a,b,c],[c,b,a])
　　=rev([a|x1],[c,b|y2])∧(x1=[b,c],y2=[a])
　　となる．この例の場合4*4=16通りの分解がある．

単位節Uを分解するとU=V∧WとなるときVをUとWの差(U-Wと記す)と呼ぶ．例１より，

rev([a|x],[c,b|y])
　　=rev([a,b,c],[c,b,a])-(x=[b,c],y=[a])

となる．

２．２　Least General Generalization(LGG)

アトムｐ，ｑのLGGの手続きは以下の1.～4.を満たす．

1. 初期設定：$\theta=\delta=\varepsilon$
2. ｐ＝ｑならば，lgg(ｐ,ｑ,θ,δ)＝(ｐ,θ,δ)
3. ｐ≠ｑでｐとｑが複合項r($s_1,\cdots,s_n$)，r($t_1,\cdots,t_n$)ならば再帰的に定義される．
　　lgg(ｐ,ｑ,θ_1,δ_1)＝p(lgg$_1$(s_1,t_1,θ_1,δ_1),…
　　lgg$_1$(s_n,t_n,θ_n,δ_n))
　　ここで，lgg$_k$(s_i,t_i,θ_i,δ_i),k∈{1,2,3}でkはlggの値のk番目の要素である．また，θ_{i+1}=lgg$_2$(s_i,t_i,θ_i,δ_i)，δ_{i+1}=lgg$_3$(s_i,t_i,θ_i,δ_i)(1≦i≦n-1)
4. その他の場合 lgg(ｐ,ｑ,θ,δ)=(ｘ,θ',δ')
　　ここで，ｘはｘθ'＝ｐ，ｘδ'＝ｑとなる代入で$\theta'=\theta\cup\{x/p\}$，$\delta'=\delta\cup\{x/q\}$となる．ｘがすでに$\theta$，$\delta$で定義されているならば$\theta'=\theta$，$\delta'=\delta$，そうでなければｘは$\theta$と$\delta$で定義されていない変数である．

３．　相関による例の解析

相関は入出力例中の任意の一対の単位節からこれに含まれる再帰的構造をLGGを用いて解析する方法であり，あるアトムの内部に他のアトムの構造が含まれていることを検出する．

【定義】　アトムp($t_1,\cdots,t_n$)の部分アトムはp($t_1',\cdots,t_n'$)である．ここで各々t_i' はt_iの部分項である．

［例２］　アトムp([[a],b],[c])の部分アトムは次のようになる．

p([[a],b],[c]),p([a],[c]),p(a,[c]),p([b],[c]),
p(b,[c]),p([],[c]),p([[a],b],c), …

《相関アルゴリズム》

入力：任意のアトムp(S),p(T)

出力：以下で定義される４組(D,p(U),E,r)の集合

手続き：各々p(T)の部分項p(T')についてLGG lgg(p(S),p(T'))＝(p(U),θ,δ)を計算する．出力される４組は以下で定義される．

1. front difference D ： $T-T'$
2. 共通部分アトム p(U) はLGGの最初の要素
3. back difference E は代入：
　　{x/t ∈ δ | tは複合項(変数でも定数でもない)}

SYNAPS：Automatic Synthesis System of Logic Programs based on Correlation of Samples

Yoshiyuki SAITO, Katsuhiko NAKAMURA

Tokyo Denki University

4. similarity factor r : f - v
ここでf はp(U)の変数と定数の数，v はp(U)の変数と定数の種類の数である．

［例３］ append/3
append([a],[b],[a,b])
append([c,d],[e],[c,d,e])
に相関を行なう．結果は3*2*4=24通りあるがsimilarity factor(r)が大きいものは以下のものである．

1. D＝append([C|X],Y,[C|Z])
p(U)＝append([x1],[y1],[x1,y1])
E＝{}　r＝4(7-3)
2. D＝append([c,d|X],Y,[c|Z])
P(U)＝append(X1,[Y1],[Z1,Y1])
E＝{}　r＝2(6-4)
3. D＝append([c|X],[e|Y],[c|Z])
p(U)＝append([x1],y1,[x1,z1])
E＝{}　r＝2(6-4)
4. D＝append(X,Y,[c|Z])
p(U)＝append([x1|x2],[y1],[z1,y1])
E＝{x2/[d]}　r＝2(7-5)

4．　相関に基づく節の合成

入出力例に相関を行なった結果からHorn節を生成するオペレータについて述べる．SYNAPSでは相関の結果(D,p(U),E,r)のE(back difference)の値によりオペレータの適用を決定する．E＝{}ならばextended absorptionオペレータ(EAオペレータ)，E≠{}ならばextended intra-constructionオペレータ(EICオペレータ)を適用しHorn節を生成する．

4．1　EA オペレータ

相関の結果(D,p(U),{},r)から次の形式の規則を生成する．

p(X)← D ∧ p(U)

これをbase ruleと呼ぶ．

absorptionオペレータ[3]はLGGにより一般化された２つのアトムからp(T)←p(S)形式の再帰的規則を生成する．EAオペレータは base rule p(X)← D ∧ p(U)のDを一般化することにより同様の規則を生成する．

《EAオペレータ》
入力：similarity factorが最大である相関の集合
出力：p(T)←p(S)形式の規則
手続き：各相関に対して以下を適用する．

1. base ruleの生成
2. front difference の一般化

［例４］ delete/3
入力：{(delete(X,[1|Y],[1|Z]),delete(A,[B,A],[B]),{},3),(delete(X,[m|Y],[m|Z]),delete([B,A,|C],[B|C]),{},3)}

1. delete(X,[1|Y],[1|Z]) ← delete(X,Y,Z)
delete(X,[m|Y],[m|Z]) ← delete(X,Y,Z)
2. delete(X,[A|Y],[A|Z])

出力：delete(X,[A|Y],[A|Z])←delete(X,Y,Z)

4．2　EIC オペレータ

intra-constructionオペレータ[3]は新しい述語qを含む p(T)←p(S),q(V) 形式の規則を生成する．EICオペレータは相関の結果(D,p(U),E,r)からp(T)←q(V)，p(S)，p(T)←p(S),q(V)形式の規則を生成する．

《EIC オペレータ》
入力：相関の結果の集合
出力：新しい述語を含んだ規則
手続き：各相関に対して以下を引き数ごとに繰り返す．

1. front differenceがあり，LGGの結果がback differenceと無関係ならばbase ruleのまま．
2. front differenceとback differenceの一般化
3. back differenceとLGGの解析を行ない新しい述語を追加

［例５］ rev/2
入力：{(rev([g|X],Y),rev([A,B],[B,A|C]),{C/[g]},2),(rev([c|X],Y),rev([A],[A|B]),{B/[c]},1)}

1. rev([_|X],_) ← rev(X,_) ,…
2. rev([L|X],_) ← rev(X,_) ,…
3. Y=[B,A|C] {C/[L]}　,Y=[A|B] {B/[c]}より，
Y=[Y1|L]
新しい述語n_p/3：n_p([L],Y1,Y)

出力：rev([L|X],Y) ← rev(X,Y1),n_p([L],Y1,Y).

5．結論

相関を用いて論理プログラムを合成することにより以下のことが明らかになった．

・帰納推論オペレータの適用条件が明確．
・similarity factorを評価値とすることで効率よく論理プログラムを合成できる．
・相関でfront differenceとback differenceを解析することにより，合成できる節の形式が明らか．

［参考文献］

[1]中村:Induction of Logic Programs based on Correlation Samples,1992
[2]G.D.Plotkin:A Note on Inductive Generalization, Machine intelligence 5,pp.153-163,1970
[3]S.Muggleton,W.Buntine:Machine Invention of First-Order Predicates by Inverting Resolution, Fifthinternational Conference on Machine Learning,pp.339-352,1988
[4]J.W.Lloyd:Foundations of Logic Programming, Springer-Verlag,1984
[5]C.Rouveirol,J.E.Puget:A Simple Solution for Inverting Resolution,Proc.of 4th European Working Session on Learning,pp.201-211,1989