人　工　知　能　105－4
（1996．8．26）

有機的プログラム： 文脈に厚いシステムにむけて

有馬　淳

株）富士通研究所
arima@iias.flab.fujitsu.co.jp

伝統的なAI研究では、知るべきすべてがあらかじめ判明しているとの仮説–閉世界仮説–の下で多くの研究がなされてきた。近年、問題解決の最中にも条件が変化しうる開世界のもとでの問題解決を研究しようとする動きが急速に起こってきた。「文脈に厚いシステム」は、問題解決の環境変化に柔軟に対処する能力において高いシステムを表し、本研究ではそういったシステムの基本的な枠組に関して幾つかの提案を行なう。

本稿ではまず、AIの従来研究–問題解決、計画問題、プログラム合成など–を統合する見方を示す。次にその見方の下で、同一の機能を持つ複数の計算主体が要求に応じて自己組織化し問題を解決するための一つのモデルを示す。

Toward Context-Thick System – Organism of Homogeneus Agents Model

Jun Arima

FUJITSU Labs. LTD.

In traditional AI researches, a concept of *closed* world assumption was one of the most common assumptions where every critical thing is known a priori. Recently, there is a rapid shift of interests of AI researchers to a problem solving under the *open* world where its important conditions possibly change in the middle of problem solving. A *context-thick* system is one of the objectives along this direction and is a robust system for variations of its surrounding environment.

This paper first introduces a unified view of problem solving, planning and program synthesis which have been explored separately in the traditional researches. Then, under this view, we propose a preliminary model in which homogeneous computational agents, given a demand, interact each other, organize others dynamically and try to meet the demands.

1　動機

「文脈（状況）に厚いシステム [4]」の重要性が昨今の open な環境下での問題解決の必要性から認識され始めている。従来のいかなる計算システムも、文脈に対して鋭敏な脆弱さを持つ。すなわち、これらのシステムは限定された範囲（文脈）の入力に対してのみ正しく動作するよう設計されており、この範囲を逸脱した入力は無意味であり危険なものであった。これに対し、そうした文脈の限定を前提としない柔軟なシステム作りを目指そうとする動きがＡＩの分野で起こっている。しかしながら、この動きはまだ始

まったばかりであり、方法論ばかりか課題の規定すらもこれからのものである。本論文ではこの動きに呼応し計算論的な立場から一つのモデルを提案し、文脈の厚さを実現するシステムについて議論したい。

想定範囲外の入力に対して耐性をもつプログラムは明らかに新たな文脈に対して対応可能な機能を自らの中に創発する能力が必要である。この創発性を持たせる課題に対し、戦略やプログラムなどの情報を遺伝的操作に基づいて交換、変化させる方法（GP[6, 3]）、試行錯誤の結果、良い結果を生んだ行動列を強化してゆく方法（強化学習 [11]）、あらかじめ用意された異質な (heterogeneous) 機能の組合せを変え、動的に組織化することによってエージェンシーとして創発を行なう方法（有機的プログラミング [9]）などのアプローチがありえる。前二者の方法において創発、適応の発現は個体にあるのに対し、後者では組織、個のつながりにあることを注目すべきである。そこでここでは特に後者の創発性を組織創発性もしくは**有機的創発性**と呼び、他のアプローチと区別する。

ここでの関心は有機的プログラミングにおける中島の捉え方により近い。但し、相同的な動作主体（エイジェント、セル）が相互作用によって異質な動作主体へ変化しそれらが組織化することで有機的創発性を生み出すことに興味の主眼がある。有機的創発を起こす機構を根源的に考えてゆくことにより、文脈に対して厚いシステムに対する理解がより深まると考えるからである。

2 統一的な見方

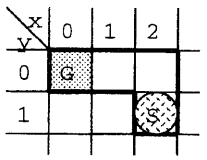
知的システムを一般的にモデル化するために、生物の活動や伝統的な AI システムを共通の枠で捉えてみたい。

動作原理 – 目標差分吸収

知的なシステムは常に目標を生成し達成しようとする試みの中に存在する。生物にとっては「知的」は生物の環境適応能力の一つの程度状態を形容するものであって、この<目標–達成>活動は生物が生存してゆく上で根源的なものと見ることができるだろう。生物の場合、不愉快な状態（現状態）から愉快な状態（目標状態）への変化、現状態と目標状態の差分吸収が自律的動作となる。言うまでもなく、この愉快、不愉快は多分に生物的なものであって、人工的な（人間に使役する）知的システムを考える場合には、システムの外から何らかの方法で人間が規定することになる。

多くの伝統的な AI システムの活動もまた、現状態から目標状態の差分吸収 [10] として見ることが可能である。プラニングや問題解決は現状態から目標状態へ遷移可能なオペレーションの列を見つける問題であり、定理証明は、現状態に対比される公理の集合から、目標状態へ対比される証明すべき定理を導く推論規則の適用列を見つけることである。この見方をここでは問題解決の文脈を越えさらに一般化する。プログラム合成は、入力（現状態）から出力（目標状態）を生成するコマンド列（プログラム）を合成することととらえることが可能である。つまり、プランング、問題解決、定理証明は 1 つの差分を吸収するオペレーターの合成を、プログラム合成は複数の差分（正例／負例）を整合的に吸収する汎用のオペレータ合成と見ることが可能である。（図 1 参照）

そこで本研究では動作主体に (現状; 目標) が与えられることを前提にし、動作主体はその<差分を吸収する>ことを自律的に行なう計算モデルを考える。(現状; 目標) は**差分環境** と呼び、ある共通のメカニズムで吸収すべき差分環境の集合を**文脈**と言うことにする。各動作主体は**外界**に分散し、互いに差分環境を外世界に生み出すことで相互反応で



int. state goal input output

S = <2,1> G = <0,0> (<2,1> ; <0,0>)

operators functions

delete add

up: <X,Y> -> <X,Y-1> λ X,Y [<X,Y-1>]

lt : <X,Y> -> <X-1,Y> λ X,Y [<X-1,Y>]

G = lt(lt(up(S)))

プラニングとプログラム合成は同じ見方がとれる。

図 1: プラニングとプログラム合成の同型性

きる。また、動作主体に関しては 中島 [9] にならい有機的な言葉使いを行う。エージェントは**細胞**とよび、各細胞は、現状態と目標状態との差分を吸収することを根源的な目的とする（差分駆動)。また、差分吸収は、現状態から目標状態を構成するメカニズムを作ることであり、メカニズムは関数の合成でモデル化される。この差分吸収は、細胞レベルの代謝系の適応的構成に類比し、基本関数を**酵素**、基本関数またはその合成関数を**代謝系**と呼ぶ。差分吸収は従来 AI で研究されてきた planning や プログラム合成を統一的に扱うことを意図すると同時に、差分駆動は生物とのアナロジーでは細胞の能動的性質を表し、自己保全や自己実現欲求に継るものと考えたい。

3 有機的相同体計算モデル

本論文では一般に十分な計算能力を持つ原始帰納関数に属する機能を単一の機能を持つエージェントの総体で実現するモデル **有機的相同体計算モデル** を考える。

外界 W： 細胞間の仮想的通信路。文脈 ($\Gamma(l)$) の集合。

細胞 A_i： 核と内界 (K_i) からなる。

内界 K_i： 代謝系の集合。代謝系は酵素と呼ばれる基本関数または代謝系から作られる関数である。但し、酵素はすべての定数関数、successor 関数、すべての identity 関数 ($\lambda X_1, \cdots, X_n.(X_i)\ \ 1 \leq i \leq n$) からなる。以下で定数は $\{0, \mathtt{true}, \mathtt{false}\}$ さらに簡単のため原始帰納関数である述語 $=, >$ の特徴関数と各ブール演算子は内界に存在するものとする。(原始帰納関数については [7] などを参考されたい。)

代謝系 ： ここでは以下の形の集合で表す。

$f(X) \longleftarrow \underline{\mathtt{if}}\ \varepsilon(X)\ \underline{\mathtt{then}}\ \alpha(X)$

文脈 $\Gamma(l)$： ラベルがついた差分環境の列。ラベル (l) は代謝系を一意に示す。同じ文脈に属する差分環境は同じ代謝系によって吸収されることが期待される。

差分環境 ： 現状態 (S) と目標状態 (G) の 2 項組 $(x;y)$。x は 項の n 項組 $x_1,\cdots,x_n$ を一般に表す $(n \geq 0)$。変数が現れない差分環境は、**強制的** であると言い、第 2 項が変数である差分環境は **随意的**であると言う。強制的差分環境 $< x;y >$ に対し、関数 f を x に作用させた結果が y である時、関数 f は差分環境$< x;y >$ に**適合する**という。この時、差分環境 $< x;y >$ は関数 f に関して**正** であるという。逆に適合しない差分環境は関数 f に関して**負**の差分環境である。随意的差分環境 $< x;Y >$ には任意の関数が適合する。文脈 Γ は差分環境の集合である。強制文脈 Γ は符合つき（$+$ または $-$）の強制的差分環境の集合である。関数 f が強制文脈Γ の各正負差分環境 $\pm e$ に対し、$+e$ の時 e に適合し、$-e$ の時 e に適合しない時、関数 f は 強制文脈 Γ に適合するという。

差分環境 $(x_1,\cdots,x_n;y)$ が差分環境 $(x'_1,\cdots,x'_n;y')$ にδ連接するとは、ある i $(1 \leq i \leq n)$ に対して、$x_1 = x'_1,\cdots,x_i = s(x'_i),\cdots,x_n = x'_n$ であることを言い、$x'_1,\cdots,x'_n = \delta(x_1,\cdots,x_n)$ で表す。また、第 i 引数は **index 引数**と呼ぶ。ここで s は successor 関数を表す。

文脈Γ において、関数 f_1, f_2 が適合する差分環境 $(\in \Gamma)$ の集合をそれぞれ E_1, E_2 とする。もし $E_1 \supset E_2$ ならば関数 f_1 は関数 f_2 よりも差分文脈 Γ に関して**厚い** (thicker) という。

核 N_i ： 表 1の協調手続きに従って動作する。

十分な差分環境が与えられれば、この手続きによって原始帰納関数クラスの関数をすべて作ることができることはあきらかであろう。

有機的代謝創発モデルでは、ユーザーも一つの細胞として捉えられ、複数存在することがありえる。ユーザーは強制的文脈を（発生し）外界に放出し、ラベルの具体化によってシステムのその文脈への適合を知ることができる。この具体化したラベルつきの随意差分環境を放出することでユーザーは同じ文脈上の新たな現状態から目標状態を得ることが可能になる。

4 考察

他の研究との関係

マルチエージェント研究との比較： 本モデルは以下の点において特徴的である。

1) 組織形成はある特殊な高機能の個体が計画立案し組織するのではないし、また、分散化した知的な個体が高度で協調的な推論を行なうことで組織されるのもでもない。各個体の動きは単純である。各個体が行なうことは、差分環境を吸収しようとすることだけであり、吸収が十分でない時に新しい差分環境を生み出し放出することだけである。この自律的な活動のみによって結果的に協調関係と高い機能（原始帰納関数クラス）が実現できる可能性を示している。

2) さまざまな機能がさまざまな差分環境によって蓄積されるため、この組織体にとってはある時点での失敗は必ずしも永久的な失敗を意味しない。

機能構成 Step:

i) 外界より文脈 $\Gamma(F,e)$ を取り込む。

例） $\Gamma(F,\texttt{true}) = \{(4;24),(3;6),(2;2),(1;1)\}$

ii) 以下の3作用のいづれかによって α を generate する。

基礎 (base fuction):

Generate $f \in K_i$ s.t. $\Gamma(F,e)$ に関し最も厚い関数;

$\alpha = f$ とおく。

連鎖 (composition):

i) $b_1,\cdots,b_n \in K_i$ を適当に選び、新たに次の文脈を作り、外界に放出。

文脈 $\Gamma(B,\texttt{true}) = \{(b_1(x),\cdots,b_n(x);y) | \ (x;y) \in \Gamma(F,e)\}$

ii) B が具体化 $(B=b)$ した時、 $\alpha(X) = b(b_1(X),\cdots,b_n(X))$ とおく。

回帰 (recurring):

i) ある差分環境 $(x,y) \in \Gamma(F,e)$ をとる。;

(x,y) にδ隣接する差分環境を有する無矛盾な文脈 $\Gamma'(F',e')$ をとる（$\Gamma'(F',e') = \Gamma(F,e)$ の場合も含む)。;

新たに、以下の文脈を作り、外界に放出。

文脈 $\Gamma(G,\texttt{true}) = \{(y',x;y) | (x;y) \in \Gamma(F,e),(x';y') \in \Gamma'(F',e'),x' = \delta(x)\}$

ii) G が具体化 $(G=g)$ した時、 $\alpha(X) = g(F(\delta(X)),X)$ および $F = F'$ とおく。

例） $\Gamma(G) = \{(6,4;24),(2,3;6),(1,2;2)\}$

代謝創発 Step (emergence of metabolism)：

発動条件 (envoke condition)：α の発動条件 ε を以下の手順で求める。

i) α が適合する$\Gamma(F,e)$ の差分環境を C^+、適合しない$\Gamma(F,e)$ の差分環境を C^- とすると、以下の文脈を外界にを放出。

文脈 $\Gamma(E,\texttt{true}) = \{(x;\texttt{false}) | (x;y) \in C^+\} \cup \{(x;\texttt{true}) | (x;y) \in C^-\}$;

ii) E が具体化 $(E=e')$ した時、 $\varepsilon(X) = e(X) \wedge \neg e'(X),\ \varepsilon'(X) = e(X) \wedge e'(X)$ とおく。

代謝定着

$F(X) \Longleftarrow$ **if** $\varepsilon(X)$ **then** $\alpha(X)$ を内界 K_i に蓄積;

If $C^- \neq \phi$

Then $C'(F,\varepsilon') = C^-$ を外界に放出。

Else 新しい名前 $newf$ を F に代入。

例） $fact(X) \Longleftarrow$ if $X=1$ then 1. $fact(X) \Longleftarrow$ if $X>1$ then $mult(fact(X-1),X)$.

表 1: 協調適応手続き

ILP 研究との比較： 本研究は論理的要素を排除した手法であるが、Mobal[5], ARDEX [1, 2] など一部の高階型のものと関係がある。但し、ILP 研究において分散環境下での合成を行なう研究はない。ここでは分散環境であることを利用して特に新述語発見において他の適合すべき問題を利用する特徴的な方法を提案している。新述語合成を最も困難にしている理由は、新述語に関する情報量の少なさに起因している。例えば先の $fact$ial の例において、補助述語 G に与えられる文脈から新述語を合成する例題 ($mult$iply の合成) を考えよう。第一引数を index 変数とすれば例えば $(2,3;6)$ に δ隣接する例とは $(3,3;9)$ または $(1,3;3)$、第二引数を index と見れば $(2,4;8)$ または $(2,1;2)$ となるがこれらは $fact$ に対する例からは与えられない。実際 $fact$ に関する例をいくら与えても G にδ隣接した例が含まれることはない。ここでの解決の仕方は、他の課題によって生じた文脈 (例えば $\Gamma(Pow) = \{(3,2;8),(2,2;4)\}((I,X;X^I))$ の文脈に対し、回帰＋基礎作用により生じる文脈 $\Gamma(Mult) = \{(2,4;8)\}$) を利用し、2 文脈の和で$\delta$隣接する例を見つける（再帰ステップ）というものである。

残された課題: 実現法について

この計算モデルの実現にあたっては、連鎖において最も大きな問題がある。外界へ放出する文脈、すなわち $b_1,\cdots,b_n \in K_i$ の選び方と連鎖の適用法である。ARDEX[1, 2] では n は x の長さとし、引数の選択、配置などに関する基本関数と背景知識 ($\sim K_i$) 中の一般の合成関数を区別し、合成の仕方に関して強い制限を設けた。この方法はひとつの解となりえるが、planning など合成が本質である課題を統一的に扱う方法には不十分である。本論文の統一的視点からの制御の方法を今後考えてゆく予定である。

参考文献

[1] 有馬 淳：論理プログラムの自動検索・統合・発見, 情報処理学会研究報告 95-AI-99, pp.165-174, 1995.

[2] Arima,J.: Automatic Logic Programming under Highly Redundant Background Knowledge, in Proc. of *the 5th International Inductive Logic Programming Workshop*, 1995.

[3] Ito,A. & Yano H.: The Emergence od Cooperation in a Society of Autonomous Agents, in Proc. of *International Conf. MultiAgent Systems'95*, pp.201-208, 1995.

[4] 橋田浩一：人工知能における基本的問題, 人工知能学会誌 Vol.10 No.3, pp.340-346, 1995.

[5] Kietz,J and Wrobel,S.: Controlling the Complexity of Learning in Logic through Syntactic and Task-Oriented Models, in *International Workshop on Inductive Logic Programming, ILP-91*, 1991.

[6] Kinnear,K.E.: *Advances in Genetic Programming*, A Bradford Book, MIT Press, 1994.

[7] Manna, Z.:, *Mathematical Theory of Computation*, McGraw-Hill, 1974.

[8] 中村運： 細胞の起原と進化, ライフサイエンス教養書 5, 培風館, 1982.

[9] 中島 秀之：情報統合のための有機的プログラミング, 人工知能学会誌, Vol.11, No.2, pp.27-34, 1996.

[10] Newell,A., Show,J.C. and Simon,H.A.: A variety of intelligent learning in a general problem-solver. In M.C.Yovits and C.Cameron (Eds.), *Self-organizing systems*, New York: Pergamon Press, pp 153-189, 1960.

[11] 山村 雅幸、宮崎 和光、小林 重信：エージェントの学習, 人工知能学会誌, Vol.10, No.5, pp.683-689, 1996.