Title	課題解決のプロセス・モデルとしてのプロダクション・システム
Sub Title	On the production system as a process model of problem solving
Author	小倉, 康仁(Ogura, Yasuyoshi)
Publisher	慶應義塾大学大学院社会学研究科
Publication year	1985
Jtitle	慶応義塾大学大学院社会学研究科紀要 : 社会学心理学教育学 (Studies in sociology, psychology and education). No.25 (1985.) ,p.55- 66
JaLC DOI	
Abstract	
Notes	論文
Genre	Departmental Bulletin Paper
URL	https://koara.lib.keio.ac.jp/xoonips/modules/xoonips/detail.php?koara_id=AN0006957X-00000025-0055

課題解決のプロセス・モデルとしてのプロダクション・システム

On the Production System as a Process Model of Problem Solving

小　倉　康　仁
Yasuyoshi Ogura

The rule-assessment approach, which was developed by R. S. Sieger, provided a developmental model of knowledge structure of subject performing balance scale tasks. In his model, Siegler formulated the development of an ability for problem solving as the sophistication of the decision-tree (i. e. the increase of nodes). To explain the cognitive development on balance scale tasks, D. Klahr presented a production system model in which the acccumulation of production rules corresponds the increase of nodes in Siegler's model.

Klahr's model, as total information processing model, has a power to generate the performance on the task while Siegler's model can only make logical prediction of the performance. In this study, I programed the production system interpreter to run the production system model and evaluated the meaningfulness and the possibility of the model.

てんびん課題における被験者の判断構造に関する発達段階モデルとしてR. S. Siegler のルール評価アプローチによるものがあるが，D. Klahr はこの心理学的モデルを，問題解決システムについての一般的なコンセプトであるプロダクション・システムで表現（プロセス・モデル化）した。本研究は，このプロダクション・システム・モデルを実際に動かすためのプロダクション・システム・インタープリターを作成し，かつ，知識の体制化や知識の発達・学習のモデル表現形式としての，プロダクション・システム・モデルの有用性と可能性について吟味・検討したものである。

1. 理論・モデルに関する方法論的考察

特定の心理学理論やモデルの有用性というものは，次に列挙するようないくつもの成分から成り立つ複雑なものであると思われる。まず最初に論理的一貫性（内部整合性）や厳密性一単純性といった形式的基準と，経験的検定可能性（予側の精密性と操作的定義）及び実際の経験的妥当性（データとの適合性）という実質的基準があげられる。次によりあいまいではあるが重要な基準として，一般性（他の状況への適用可能性·拡張可能性），もっともらしさ（対象領域に関する他の知識との整合性），パラダイム生産性（新たな実験パラダイムや新たな仮説を誘発する力），応用可能性（例えば教育へのインプリケーションなど）等々が考えられる。

心理学理論やモデルを計算機シミュレーションが可能な情報処理プロセス・モデルとして表現することは，上述した諸基準をクリアーするための有力な手段である。何故なら知識表現の形式化を通じて理論・モデルはより明確なものとなり，その隠れた欠陥や新たな問題点が発見されたり，有効なアイデア（新しい実験デザインや作業仮説）が考え出されたりするからである。

2. ルール・モデル

認知発達の研究においては，被験者の知的発達水準を調べるためにてんびん課題がよく使われるが，choice型

のてんびん課題に関する課題解決能力の発達を記述したモデルとして R. S. Siegler のルール・モデル（Siegler 1978を参照）がある。これは，課題解決のための手続きを2進木構造を持つ decision-rule として表現した上で，各発達段階のパフォーマンス・パターンをその段階に特有のルールによって予測・説明しようとするものである。従って知識水準の発達や学習は，あるルールからより複雑な構造を持ったルールへの移行（2進木のノードの分化）として定義される。

Siegler はより高度なルールへの移行の原動力の1つとして，エンコーディング機能の改善に注目しているが，段階間移行を説明するモデルの構築には至っていない。

3. プロダクション・システム・モデル

D. Klahr は，decision-rule の2進木構造をプロダクション・ルールの集合体で表現し，各発達段階をそれぞれプロダクション・システム（その段階に特有なプロダクション・ルール集合とエンコーディング・オペレーター集合によって特徴づけられる）でモデル化（Klahr 1980を参照）した。それによると発達段階間の移行は，プロダクション・ルールの付加・書き換え（プロダクションが必要とする情報のエンコーディング機能の改善を伴う）として定義される。こうして段階間移行は，より精密に定義されることになった。

彼はさらに，段階間移行を因果的に説明するメカニズムとして，適応プロダクション・システム（安西，佐伯，難波 1982を参照）で表現されるモデルを構想している。

3.1 プロダクション・システム

プロダクション・システムとは，一定の知識表現形式あるいは知識の体制化（organization）の枠組みのことで，問題解決用の人工知能システムなどによく使われている。システムの基本構成は，特定の作業を遂行するための手続き的知識表現であるプロダクション・ルールの集合と，事実・データ表現の集合である作業記憶とから成る。プロダクション・ルール（単にプロダクションとも言う）は「～ならば――せよ」という表現形式をとり，この前半部分を条件，後半部分をアクションと呼ぶ。

全てのプロダクション・ルールが作業記憶の内容と照合され，その条件の全てが作業記憶要素によって満たされたプロダクションのアクションが実行される。作業記憶の内容と適合するプロダクションがいくつかある場合には，あらかじめ定められた基準にのっとって，適用すべきプロダクションが1つ決定される。アクションが実行されるとその結果として，作業記憶の内容が更新（付加，消去，書き換え）される。以上のサイクルを繰り返して課題を遂行するというのが，プロダクション・システムの基本動作パターンである。

3.2 ルールⅢA

Siegler のルール・モデルにおけるルールⅢはフィードバックの処理能力を持たないが，そのような能力を持ち，かつルールⅢの方法論的難点を部分的に克服した代替案としてルールⅢA（図1参照）がある。これは，フィードバックの内容に従って CRITERION（判断における次元の優位関係を決める基準）が変化することによって，次の試行でのパフォーマンスが影響を受けるという仕組みのルールである。ルールⅢの代替案であるので，最初の3つの test nodes はルールⅢと同じになっている。Klahr は，このルールⅢAにのっとって判断を行う被験者の心理過程を，プロダクション・システム・モデルで表現した。

3.3 ルールⅢAを使うプロダクション・システム

前述のルールⅢAを体現するプロダクション・システムは，プロダクション・ルール，作業記憶，オペレーターの3つから構成される。

3.3.1 プロダクション・ルール集合

このシステムで使われるプロダクション・ルールは，判断・推論のための手続き的知識表現であり，その機能面から次の3種類に分類される（図3参照）。

(1) Pn――掛け金をはずした時にてんびんがどちらに傾くか（またはバランスするか）を予想するための手順と推論規則であり，各プロダクションが decision-tree の各ノードに対応している。

(2) En――予想の後処理の手順であり，予想と実際の動きとの比較を行ない，結果を記録する。

(3) SWn――フィードバックの後処理の手順であり，予想の正誤に従って判断基準の変更を行なう。

一方プロダクションのアクション部分に注目した場合，その作用形態から見て次の2つのタイプのアクションが使われていることがわかる。

(A) 作業記憶に単に情報を付加するアクション――これによって供給される新情報は常に作業記憶の先頭に付け加えられる。その際アクション部分での表現

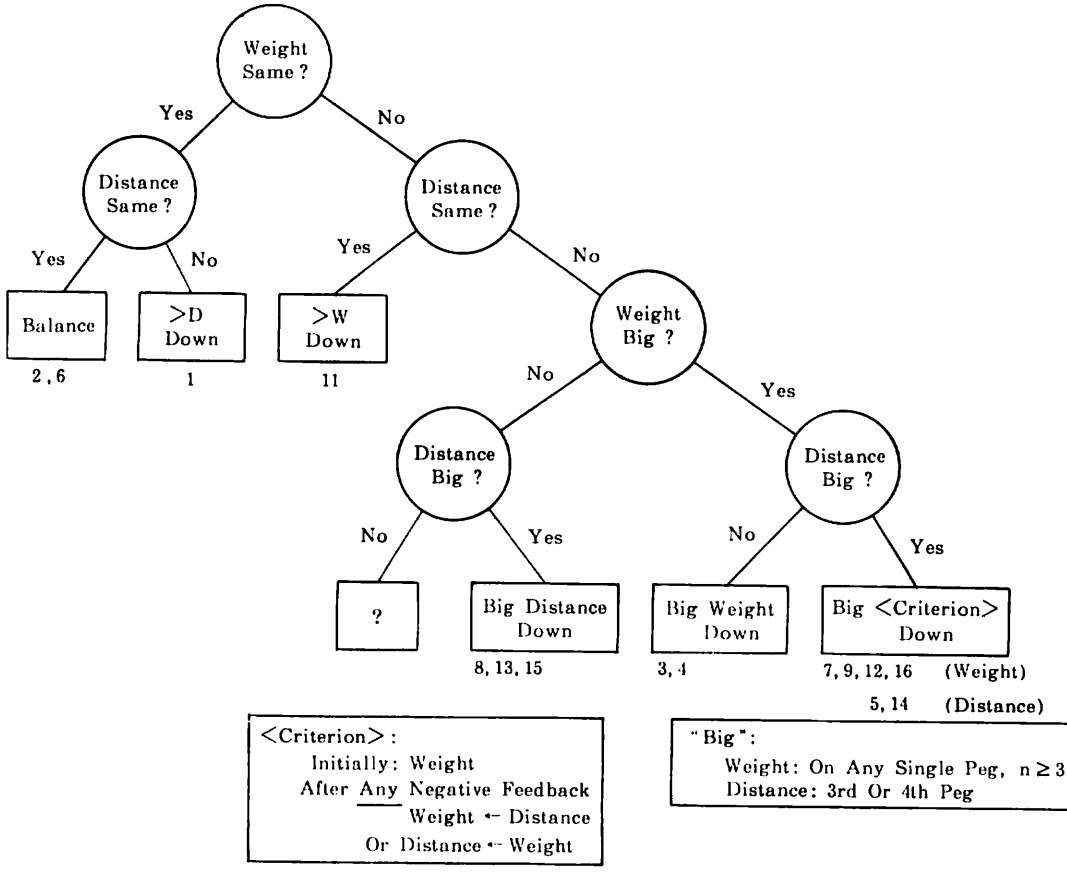

図1　rule ⅢA の decision-tree 表現

が直接に（変数に値を割り当てた形で）転写される場合と，アクション部分で呼び出されたオペレーターが働いた結果（オペレーターからの出力）として外部入力情報が書き込まれる場合とがある。

(B) 作業記憶中の既存の情報（要素）を書き換えるアクション——これには2種類あって，(A→B)というアクションは作業記憶中の第2番目の要素に含まれている表現Aを表現Bに置き換え，(X**)というアクションは作業記憶の先頭にある表現(A)を(X(A))のような修飾表現に変える。

プロダクション・ルールはその表現中に変数を含んでいるので，条件と作業記憶要素との照合や，直接転写型のアクションの実行は変数束縛を必要とする。

またプロダクションの条件部分中には，(WEIGHT MORE) とか (DISTANCE MORE) といった縮約形表現が存在するので，プロダクション条件と作業記憶要素との照合に際しては，一種のパターン・マッチングが行なわれなければならない。縮約形表現としては，この他にも (EXPECT) と (SEE) と (CRITERION) がある。

さらに，P8 の (ABS (⟨DIMENSION 1⟩)) のように，変数表現と縮約形表現が同居している場合もあるので，照合のための機構はかなり複雑になる。

ついでに説明しておくと，この (ABS (～)) のような表現をABS条件と呼び，(～) という表現と適合する作業記憶要素が存在しない時にのみ満たされる条件である。ABS条件としては，P8の他にE3の (ABS (SEE ⟨SIDE 1⟩ ⟨DIRECTION⟩)) がある。

なお，作業記憶の内容に適合するプロダクションが同時にいくつも存在する場合には，このシステムにおいては次の2つの原則 (conflict resolution principles) に従って選抜が行なわれる。1つは Special Case Order と呼ばれるもので，これによって論理的により特殊な条件を持つプロダクションが優先される。もう1つは Working Memory Order と呼ばれるもので，これに

よって，作業記憶のより先頭に近いところに並んでいる要素と適合するプロダクションが優先される。その妥当性は，最近つくられた情報及び最近使われた情報は重要度が高いはずだから優位性を持たせるべきであるという考え方に拠っている。

3.3.2 作業記憶

知識（プロダクション）の解発に必要なデータの一時的収納場所であり，問題解決のための情報処理過程で常にその内容が更新されていく。作業記憶の各要素は，プロダクションの条件やアクションと同一（共通）の構文で書かれた事実表現であって，その性格上変数は一切含んでいない。

作業記憶の内容が更新されるのは，プロダクションのアクションが実行された結果としてだけではない。適用されるべきプロダクションが決定された後で（アクションが実行される前に），そのプロダクションの条件部分と適合した作業記憶要素は全て，作業記憶の先頭に移項される。この自動的リハーサル機能は Working Memory Order が正しく機能するために不可欠のものである。

3.3.3 オペレーター

プロダクション・ルールが内的操作を表現したものであるのに対して，知覚—運動機能（環境との相互作用としての行為）を表現するものである。いわば外界とのインターフェースとして働くという重要な役割りを持っているが，各オペレーターの内容（心理学的メカニズム）はモデル化できる程解明されていない。従ってここではブラック・ボックスとして扱い，それぞれが呼び出される条件とその機能（出力）によって各オペレーターを定義した。このプロダクション・システムは以下に述べる5つのオペレーターを持っている。

(1) ATTEND——「左右どちらの腕の方がより多くのおもりをのせているか，または支点からより遠いところにおもりをのせているか」という情報，あるいは「おもりの数，または支点からおもりののっているところまでの距離が左右同一であるかどうか」という情報をエンコードする。

(2) FIND—BIG——「左右どちらの腕でおもりの数，または支点からおもりまでの距離が "BIG"（おもりの数で3つ以上，支点からの距離で3単位以上をこのカテゴリーに入れる）であるか」という情報をエンコードする。

(3) LOOK——掛け金をはずした時にてんびんが実際に「左右どちらに傾いたか」，あるいは「バランスしたか」という情報をエンコードする。

(4) SAY—B——「バランスするであろう」という予想を口に出して言う。反応生成器の役割りを果している。

(5) SAY—D——「左(または右)に傾くであろう」という予想を口に出して言う。(4)と同様の機能を持つ。

3.3.4 ルール・モデルとの差異

Siegler のルール・モデルの本体である decision-tree 表現は，判断過程の論理的構造を記述しているにすぎず，パフォーマンスの生成力を持っていない。

一方 Klahr のプロダクション・システム・モデルは，パフォーマンスの生成力を持った具体的で全体的な情報処理プロセス・モデルとなっている。何故ならば，そのモデルは，decision-rule にのっとって実際に判断（問題解決）を行なう時の逐時的心理過程を演出するために必要な全ての機能・メカニズム（知覚—運動的オペレーターや作業記憶）を取り込んで，トータル・システムとして表現したものだからである。従ってそのプロダクション・ルール表現中には，decision-rule だけではなくプロセス全体の動き（流れ）を制御するためのフェーズ情報（情報処理過中のいろいろな局面を表わす）やオペレーター・コールが含まれている。

3.4 インタープリターとシステムの動き方

ここでは，プロダクション・システムで表現されたモデルを計算機上で稼動させるためのプログラムであるインタープリターの製作と，それを使ったサンプル・ランについて解説する。なお，このプロダクション・システムとインタープリターは，NEC の PC—8001mkII の CPM 上で走る LISP 81 (by micro communications, 1981) にインプリメントされた。今回インプリメントしたプロダクション・システムとインタープリターを総称して，PSBST—1 (Production System for Balance Scale Task—1) と呼ぶことにする。

3.4.1 インタープリター

これは筆者のオリジナル・プログラムであるが，一般的なプロダクション・システム・インタープリターの仕様に，おおむね準拠していると思われる。主要関数の呼び出し関係にもとづいて，このインタープリターを分析してみると，図2に示されるような階層構造をなしていることがわかる。なお図2においては，左右が階層構造の上下関係を，上下が関数呼び出しの時間的順序を表わしている。

BS—JUDGES は最上位関数であり，PSBST—1 の大

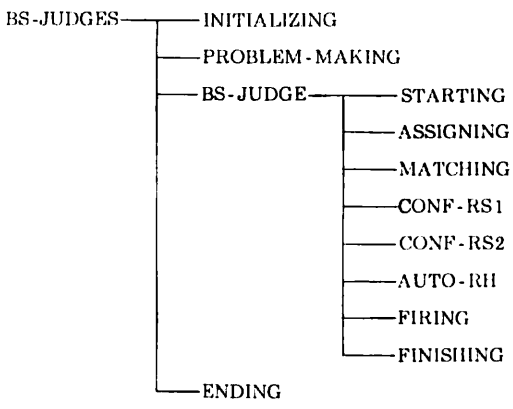

図2　インタープリターの構造

局的な動きを制御している。INITIALIZING は，作業記憶の立ち上げを行ない，CRITERIONにデフォルト値をセットする。PROBLEM-MAKING は問題入力用ルーティンであり，入力に従って問題をリセットする。入力された問題を解く問題解決器 (problem solver) の本体が，BS-JUDGEである。ENDINGは，人間がPSBST-1 に次の試行の必要がないことを伝えた時に，作業記憶をオフにして全処理を終了する (BS-JUDGESから抜け出す)。

STARTINGは，各試行での最初の目標(PREDICT＝予想せよ) を作業記憶に設定し，各フラグや各変数の初期化を行なう。ASSIGNING は，プロダクション・ルール表現中の変数に実際の値を割り当てる。MATCHING は，プロダクションの条件部分と作業記憶内容との照合を行ない，条件が満たされた全てのプロダクションをリスト・アップする。CONF-RS1 と CONF-RS2は，前述の conflict resolution principles にもとづいて，適用すべきプロダクションを1つに絞る。AUTO-RH は自動的リハーサル機能を司どり，作業記憶要素の並べ換えを行なう。FIRING は，MATCHING→CONF-RS1→CONF-RS2 という経過を経て選抜されたプロダクションのアクションを実行する。FINISHINGは，1つの試行が終了した後で作業記憶をリセットし，CRITERION情報を次の試行へ受け渡す (つまり，CRITERION情報以外の作業記憶要素を全て消去する)。

ところで，実際にインタープリターを製作してみてわかったこととして，次のようなことがあげられる。まず1つは，conflict resolution principle の具体的内容を煮つめることが，モデルの精密さや心理学的妥当性 (または，もっともらしさ)を高めることにつながるということである。そしてもう1つは，自動的リハーサル機能が，作業記憶の内容を書き換えるタイプのアクションと不可分の関係にあるということである。このように，インタープリターを実際に製作することは，特定のプロダクション・システム・モデルを理解したり，発展させたりするためのかなり有効な方法である。

3.4.2 サンプル・ラン

ここでは，PSBST-1のサンプル・ラン (出力例) について，処理の流れに沿って解説する。処理の基本的な流れは，次の5つの局面に分けられる。つまり，(a)おもりの配置の観察，(b)てんびんの動き方の予想，(c)実際の動きの観察，(d)予想と現実との比較，(e) 判断基準の変更 (もし必要ならば) の5つである。Pn タイプのプロダクションは(a)と(b)で，En は(c)と(d)で，SWn は(e)の局面で使用される。従ってプロダクションの流れとしては，Pn→En→SWnという形になる。3.4.1 の各関数の機能の説明を参考にしながら，ここの本文と図3を見ていただければ，PSBST-1 の動作の概略は理解していただけるものと思われる。

システムを動かすためには，まずプロダクション・ルール集合がセーブされているファイルをあらかじめロードしておいてから，最上位関数BS-JUDGESを評価する (図3の中の1の部分にあたる)。すると，長期記憶に保持されているプロダクション・ルールが全て表示される (2)。このシステムは，プロダクションの並べ方を気にする必要がないように設計されているので，ここでの配列は，ほぼ各々が適用される順序に従っているにすぎない。

ルール表示の直後に，CRITERIONのデフォルト値が作業記憶に入れられる。これは，最初の試行がスタートした時点で表示される (5)。この後で，PSBST-1 を次の問題に取り組ませたいかどうかをたずねてくるので，YES と答える (3)。そうすると問題の入力を要求してくるので，左腕と右腕のおもりの配置を指定してやる(4)。

トライアル1が始まると(つまり，サイクル1では)，まず目標が設定され (6)，変数への実際の値の割り当てが行なわれる (7)。そして，作業記憶の内容と適合する全てのプロダクション・ルールが，各々が適合する作業記憶要素の位置のリストを伴なった形で表示される (8A，典型例として8B)。次に，Special Case Order によるふるい落としに生き残ったプロダクション群が表示される (9A，典型例として9B)。さらに Working Memory Order によるふるい落としを通して，最終的

に単一のプロダクションが選抜され(10A,典型例として10B),アクションが実行される。その結果,ATTENDというオペレーターが呼び出され(11),その出力が作業記憶に書き加えられる。

サイクル2ではオペレーターFIND-BIGが呼び出され(13),予想をするのに必要な情報がせいぞろいする。サイクル3では,予想の内部表象が生成され(15),かつ問題に対する解答(つまり予想)が出力される(16)。サイクル4ではオペレーターLOOKが呼ばれ(17),てんびんの実際の動きが作業記憶に記録される。サイクル5では,予想と実際の動きとの比較の結果,予想は間違っていたという情報が作業記憶に付加される(18)。サイクル6では,CRITERION情報を書き換えるようなアクションを持つプロダクションが選抜—適用される(19)。その結果,トライアル1終了時における CRITERION の値は,おもりから距離へと変化している(20)。

1試行を終了後,次の試行へと進むのかどうかを尋ねてくるので,NOと答えると全ての処理を終了する(21)。

なお図3で,MATCHING FACT として表示されているものは,選抜されたプロダクションの条件部分の表現と適合する作業記憶要素であり,その適合がパターン・マッチングにもとづく場合(例えば12)と,変数束縛にもとづく場合(例えば14)とがある。また,INTERPRETATION として表示されているものは,プロダクションの直接転写型アクションの変数を含む表現を,変数束縛に従って解釈したものである(例えば15)。

図3は(0004:0020)という問題を与えてみた場合であるが,当然,他のタイプの問題に対しても ruleⅢA にのっとって反応する。従って,(0010:2000)というような問題を与えられると,!NO BIG DIMENSION IN ANY SIDE ! ! I CAN NOT SOLVE THE PROBLEM ! と応えてギブ・アップしてしまう。

4. プロダクション・システム・モデルの評価

最後に,以下に述べる3つのポイントに焦点を合わせて,プロダクション・システム・モデルの評価を行なってみたい。

4.1 方法論的観点から

running model としての Klahr のプロダクション・システム・モデルは,特定の課題解決に必要な全ての手続き的知識と心理的機能を,同質の表現で統合的にモデル化したものである。従って,そのモデル構築過程において,decision-tree 表現では視野に入ってこなかったことが,明確な問題意識となって現われてきた。例えば,プロダクションとそれが利用する情報を供給するエンコーディング機能との相互関連性の問題がそうである。また,課題解決過程の各時点における作業記憶への負荷量の理論的予測の問題(chunking の問題を含む)なども,そうしたものの1つである。

4.2 その心理学的妥当性に関して

4.2.1 発達・学習の問題

Klahr の ruleⅢA ためのプロダクション・システムモデルは,フィードバックの内容を次の試行でのパフォーマンスに反映させることができるようになっている。しかしそれは,作業記憶要素を使ったちょっとした工夫で「学習的なふるまい」をまねたにすぎず,プロダクション集合自体には何の変化も生じない。本当の意味の学習が起こるためには,自分の行為についての経験記憶と因果的推論にもとづいて誤反応の原因を究明し,手続き生成のための手続きによって新しい手続きを造り出すといった機能(学習能力)が必要である。そのような機能を具備した自己修正的なシステムは,適応プロダクション・システムでモデル化されなければならない。

ところで,相対的な定性表現しかないモデルⅢから,絶対的定量表現(数量表現)を使うモデルⅣへといきなり移行するのは不自然であり,二分法的(定性的)ながらもナイーブな絶対的表現を含むⅢAから,数量表現を駆使するⅣへと移行する(つまり,Ⅰ→Ⅱ→ⅢA→Ⅳと移行する)と考えた方が無理がない。この仮説は,心理学的妥当性の観点とともに,人工知能技法の面から見ても適切なものと思われる。その意味で,Klahr のプロダクション・システム・モデルⅢAを,発達・学習を説明するようなモデル(適応プロダクション・システム・モデル)を構築するための準備作業の1つとして位置づけることができる。

4.2.2 理解の問題

Klahr のモデルⅢAに代表されるプロダクション・システム・モデル,および適応プロダクション・システム・モデルは,状況個別的な課題遂行能力(問題解決のための手続き的知識)や,その発達・学習をモデル化することにある程度成功していると言える。それというのも,プロダクションの IF-THEN 構造と各プロダクションの独立性・断片性が,手続き的知識やその増加を表現するのに適しているからであると考えられる。

しかしながら,単に特定の課題を解決できるからといって,その課題状況を概念的に理解していることにはな

らない。状況の概念的理解とは，ある状況が他の状況（とくに類似状況）や他の概念と意味的に関連づけられている状態を指す。ある状況を，共通の構造（同一構造または同型構造）を持ちつつ表面的様相を異にする他の状況と結びつけるためには，問題の場面を実際に，あるいは心の中で動かしてみる（いじりまわして，いろいろに変形してみる）ことが必要である。

例えばてんびん課題の場合には，「わざと片方の腕のおもりを取り去ってしまってから，その腕を自分の手で抑えながら掛け金をはずしてみる」とか，「左右どちらかに傾いてしまっているてんびんを，はね上っている方の腕を上から押し下げる――さがっている方を持ち上げるのではなく――ことによって平衡状態にする」とかしてみれば，当該場面（状況）を，てこを使って物を動かそうとしている場面と結びつけることができるのではないだろうか。このような関連づけが行なわれると，てこの場面に関するエピソード記憶（感覚一運動的経験）や体験的に獲得された「てこの経験則」にもとづいて，てんびん課題を解決するための手続きが新たに生成されたり，既に使用されている手続きの意味が実感として納得され，その正当性が認識されたりする。

なお「てこの経験則」というのは，「動かそうとする物にあてがったてこの先端（作用点）から支えとなるもの（支点）までの距離が短く，支えから力を加えるところ（力点）までの距離が長い程，より少ない力で物を動かすことができる」という内容のものである。これはまだ定性的な形で定式化された知識表現にすぎないが，将来において「力のモーメント」の物理公式（定量的定式化）を理解するための有力な基盤になると考えられる。

いろいろな状況（場面）に関するエピソード（経験記憶）は主に，プロダクションのような手続き的知識表現ではなくて，宣言的知識表現によって取り扱われる。宣言的知識表現は，SCRIPTなどに代表されるように，一般的な意味要素で記述された意味単位から構成される階層構造化された意味ネットワークとして組織される。従って，単なる課題解決能力と概念的理解とのちがいを，状況個別的な手続き的知識と，構造化された宣言的知識である世界知識（knowledge of the world）との差異に対応づけて考えることもできる。

それゆえ，人間のダイナミックな知識構造や理解過程をトータルにモデル化するためには，両方の知識表現をうまく組み合わせたシステム設計が要求されるものと思われる。そのようなシステムで表現されたプロセス・モデルをもってしてはじめて，全体的な知識構造の動き（発達・学習）を記述一説明することが可能になるであろう。

文　献

1) 安西祐一郎，佐伯　胖，難波和明　LISPで学ぶ認知心理学2　問題解決　東大出版会，1982
2) Klahr, D. Information-processing models of intellectual development. In R. H. Kluwe & H. Spada (Ed.), Developmental models of thinking. Academic Press, 1980
3) Klahr, D. & Wallace, J. G. Cognitive development: An information processing view. Lawrence Erlbaum Associates, 1976
4) Siegler, R. S. The origins of scientific reasoning. In R. S. Siegler (Ed.), Children's thinking: What develops?. Lawrence Erlbaum Associates, 1978
5) Wilkening, F. & Anderson, N.H. Comparison of two rule-assessment methodologies for studying cognitive development and knowledge structue. Psychological Bulletin, 1982, 92-1
6) Winston, P. H. Artificial intelligence. Addison-Wesley Publishing, 1977

```
(1) >(BS-JUDGES)

    ### THE PROCESS START ###

    ::: RULES IN LONG TERM MEMORY (LTM) :::
    ((P8 (IF (PREDICT) (ABS (<DIMENSION1>))) (THEN ATTEND))
     (P1 (IF (PREDICT) (WEIGHT SAME))
         (THEN (MADE **) (EXPECT BOTH LEVEL) SAY-B) )
     (P2 (IF (PREDICT) (WEIGHT MORE <SIDE1>))
         (THEN (MADE **) (EXPECT <SIDE1> DOWN) SAY-D) )
     (P3 (IF (PREDICT) (WEIGHT SAME) (DISTANCE MORE <SIDE1>))
         (THEN (MADE **) (EXPECT <SIDE1> DOWN) SAY-D) )
     (P4 (IF (PREDICT) (WEIGHT MORE) (DISTANCE MORE)) (THEN FIND-BIG))
     (P5 (IF (PREDICT)
             (CRITERION <DIMENSION1>)
             (<DIMENSION1> BIG <SIDE1>)
             (<DIMENSION2> BIG <SIDE2>) )
         (THEN (MADE **) (EXPECT <SIDE1> DOWN) SAY-D) )
(2)  (P6 (IF (PREDICT) (WEIGHT BIG <SIDE1>))
         (THEN (MADE **) (EXPECT <SIDE1> DOWN) SAY-D) )
     (P7 (IF (PREDICT) (DISTANCE BIG <SIDE1>))
         (THEN (MADE **) (EXPECT <SIDE1> DOWN) SAY-D) )
     (E1 (IF (EXPECT)) (THEN LOOK))
     (E2 (IF (EXPECT <SIDE1> <DIRECTION>) (SEE <SIDE1> <DIRECTION>))
         (THEN (DID **) (SEE => SAW) (RESULT CORRECT)) )
     (E3 (IF (EXPECT <SIDE1> <DIRECTION>)
             (ABS (SEE <SIDE1> <DIRECTION>))
             (SEE) )
         (THEN (DID **) (SEE => SAW) (RESULT WRONG)) )
     (SW1 (IF (RESULT WRONG) (CRITERION DISTANCE))
          (THEN (OLD **) (DISTANCE => WEIGHT)) )
     (SW2 (IF (RESULT WRONG) (CRITERION WEIGHT))
          (THEN (OLD **) (WEIGHT => DISTANCE)) )
     (SW3 (IF (RESULT CORRECT) (CRITERION)) (THEN (OLD **))) )

(3) GO TO NEXT PROBLEM ?
    YES

    PLEASE INPUT WEIGHT CONFIGURATION IN LEFT HAND !
(4) (1 4)

    PLEASE INPUT WEIGHT CONFIGURATION IN RIGHT HAND !
    (3 2)

    ## TRIAL 1 START ##

(5) :: WORKING MEMORY AT START OF TRIAL 1 ::
    ((CRITERION WEIGHT))

    # CYCLE 1 START #

(6) : WORKING MEMORY AT START OF CYCLE 1 :
    ((PREDICT) (CRITERION WEIGHT))

(7) +++ ASSIGNING DONE +++
    <DIMENSION1> = WEIGHT
    <DIMENSION2> = DISTANCE

(8A) = CONFLICT SET AFTER MATCHING =
     ((1 15) P8)

(9A) = CONFLICT SET AFTER SPECIAL CASE ORDER =
     ((1 15) P8)

(10A) = SELECTED RULE AFTER WM ORDER =
      P8
```

図 3 - (1)

```
+++ MATCHINGFACT OF (PREDICT) +++
(PREDICT)

: WORKING MEMORY AFTER AUTOMATIC REHEARSAL :
((PREDICT) (CRITERION WEIGHT))

+++ OPERATOR ATTEND CALLED +++

+++ OUTPUT FROM ATTEND ( INPUT TO WM ) +++
((DISTANCE MORE RIGHT) (WEIGHT MORE LEFT))

: WORKING MEMORY AFTER FIRING :
((DISTANCE MORE RIGHT) (WEIGHT MORE LEFT) (PREDICT) (CRITERION WEIGHT))

: WORKING MEMORY AT END OF CYCLE 1 :
((DISTANCE MORE RIGHT) (WEIGHT MORE LEFT) (PREDICT) (CRITERION WEIGHT))

# CYCLE 1 END #

# CYCLE 2 START #

: WORKING MEMORY AT START OF CYCLE 2 :
((DISTANCE MORE RIGHT) (WEIGHT MORE LEFT) (PREDICT) (CRITERION WEIGHT))

+++ ASSIGNING DONE +++
<SIDE1> = LEFT
<SIDE2> = RIGHT

= CONFLICT SET AFTER MATCHING =
((3 2) P2)
((3 2 1) P4)

= CONFLICT SET AFTER SPECIAL CASE ORDER =
((3 2 1) P4)

= SELECTED RULE AFTER WM ORDER =
P4

+++ MATCHINGFACT OF (PREDICT) +++
(PREDICT)

+++ MATCHINGFACT OF (WEIGHT MORE) +++
(WEIGHT MORE LEFT)

+++ MATCHINGFACT OF (DISTANCE MORE) +++
(DISTANCE MORE RIGHT)

: WORKING MEMORY AFTER AUTOMATIC REHEARSAL :
((PREDICT) (WEIGHT MORE LEFT) (DISTANCE MORE RIGHT) (CRITERION WEIGHT))

+++ OPERATOR FIND-BIG CALLED +++

+++ OUTPUT FROM FIND-BIG ( INPUT TO WM ) +++
((DISTANCE BIG RIGHT) (WEIGHT BIG LEFT))

: WORKING MEMORY AFTER FIRING :
((DISTANCE BIG RIGHT) (WEIGHT BIG LEFT) (PREDICT) (WEIGHT MORE LEFT) (DISTANCE M
ORE RIGHT) (CRITERION WEIGHT))

: WORKING MEMORY AT END OF CYCLE 2 :
((DISTANCE BIG RIGHT) (WEIGHT BIG LEFT) (PREDICT) (WEIGHT MORE LEFT) (DISTANCE M
ORE RIGHT) (CRITERION WEIGHT))

# CYCLE 2 END #

# CYCLE 3 START #
```

図 3 - (2)

```
: WORKING MEMORY AT START OF CYCLE 3 :
((DISTANCE BIG RIGHT) (WEIGHT BIG LEFT) (PREDICT) (WEIGHT MORE LEFT) (DISTANCE M
ORE RIGHT) (CRITERION WEIGHT))

= CONFLICT SET AFTER MATCHING =                                    (8B)
((3 4) P2)
((3 4 5) P4)
((3 6 2 1) P5)
((3 2) P6)

= CONFLICT SET AFTER SPECIAL CASE ORDER =                          (9B)
((3 4 5) P4)
((3 6 2 1) P5)

= SELECTED RULE AFTER WM ORDER =                                   (10B)
P5

+++ MATCHINGFACT OF (PREDICT) +++
(PREDICT)

+++ MATCHINGFACT OF (CRITERION <DIMENSION1>) +++                   (14)
(CRITERION WEIGHT)

+++ MATCHINGFACT OF (<DIMENSION1> BIG <SIDE1>) +++
(WEIGHT BIG LEFT)

+++ MATCHINGFACT OF (<DIMENSION2> BIG <SIDE2>) +++
(DISTANCE BIG RIGHT)

: WORKING MEMORY AFTER AUTOMATIC REHEARSAL :
((PREDICT) (CRITERION WEIGHT) (WEIGHT BIG LEFT) (DISTANCE BIG RIGHT) (WEIGHT MOR
E LEFT) (DISTANCE MORE RIGHT))

+++ INTERPRETATION OF (EXPECT <SIDE1> DOWN) +++                     (15)
(EXPECT LEFT DOWN)

+++ OPERATOR SAY-D CALLED +++                                      (16)

########## LEFT DOWN

: WORKING MEMORY AFTER FIRING :
((EXPECT LEFT DOWN) (MADE (PREDICT)) (CRITERION WEIGHT) (WEIGHT BIG LEFT) (DISTA
NCE BIG RIGHT) (WEIGHT MORE LEFT) (DISTANCE MORE RIGHT))

: WORKING MEMORY AT END OF CYCLE 3 :
((EXPECT LEFT DOWN) (MADE (PREDICT)) (CRITERION WEIGHT) (WEIGHT BIG LEFT) (DISTA
NCE BIG RIGHT) (WEIGHT MORE LEFT) (DISTANCE MORE RIGHT))

# CYCLE 3 END #

# CYCLE 4 START #

: WORKING MEMORY AT START OF CYCLE 4 :
((EXPECT LEFT DOWN) (MADE (PREDICT)) (CRITERION WEIGHT) (WEIGHT BIG LEFT) (DISTA
NCE BIG RIGHT) (WEIGHT MORE LEFT) (DISTANCE MORE RIGHT))

= CONFLICT SET AFTER MATCHING =
((1) E1)

= CONFLICT SET AFTER SPECIAL CASE ORDER =
((1) E1)

= SELECTED RULE AFTER WM ORDER =
E1
```

図 3 - (3)

```
+++ MATCHINGFACT OF (EXPECT) +++
(EXPECT LEFT DOWN)

: WORKING MEMORY AFTER AUTOMATIC REHEARSAL :
((EXPECT LEFT DOWN) (MADE (PREDICT)) (CRITERION WEIGHT) (WEIGHT BIG LEFT) (DISTA
NCE BIG RIGHT) (WEIGHT MORE LEFT) (DISTANCE MORE RIGHT))

+++ OPERATOR LOOK CALLED +++

+++ OUTPUT FROM LOOK ( INPUT TO WM ) +++
((SEE RIGHT DOWN))

: WORKING MEMORY AFTER FIRING :
((SEE RIGHT DOWN) (EXPECT LEFT DOWN) (MADE (PREDICT)) (CRITERION WEIGHT) (WEIGHT
 BIG LEFT) (DISTANCE BIG RIGHT) (WEIGHT MORE LEFT) (DISTANCE MORE RIGHT))

: WORKING MEMORY AT END OF CYCLE 4 :
((SEE RIGHT DOWN) (EXPECT LEFT DOWN) (MADE (PREDICT)) (CRITERION WEIGHT) (WEIGHT
 BIG LEFT) (DISTANCE BIG RIGHT) (WEIGHT MORE LEFT) (DISTANCE MORE RIGHT))

# CYCLE 4 END #

# CYCLE 5 START #

: WORKING MEMORY AT START OF CYCLE 5 :
((SEE RIGHT DOWN) (EXPECT LEFT DOWN) (MADE (PREDICT)) (CRITERION WEIGHT) (WEIGHT
 BIG LEFT) (DISTANCE BIG RIGHT) (WEIGHT MORE LEFT) (DISTANCE MORE RIGHT))

+++ ASSIGNING DONE +++
<DIRECTION> = DOWN

= CONFLICT SET AFTER MATCHING =
((2) E1)
((2 15 1) E3)

= CONFLICT SET AFTER SPECIAL CASE ORDER =
((2 15 1) E3)

= SELECTED RULE AFTER WM ORDER =
E3

+++ MATCHINGFACT OF (EXPECT <SIDE1> <DIRECTION>) +++
(EXPECT LEFT DOWN)

+++ MATCHINGFACT OF (SEE) +++
(SEE RIGHT DOWN)

: WORKING MEMORY AFTER AUTOMATIC REHEARSAL :
((EXPECT LEFT DOWN) (SEE RIGHT DOWN) (MADE (PREDICT)) (CRITERION WEIGHT) (WEIGHT
 BIG LEFT) (DISTANCE BIG RIGHT) (WEIGHT MORE LEFT) (DISTANCE MORE RIGHT))

+++ INTERPRETATION OF (RESULT WRONG) +++
(RESULT WRONG)

: WORKING MEMORY AFTER FIRING :
((RESULT WRONG) (DID (EXPECT LEFT DOWN)) (SAW RIGHT DOWN) (MADE (PREDICT)) (CRIT
ERION WEIGHT) (WEIGHT BIG LEFT) (DISTANCE BIG RIGHT) (WEIGHT MORE LEFT) (DISTANC
E MORE RIGHT))

: WORKING MEMORY AT END OF CYCLE 5 :
((RESULT WRONG) (DID (EXPECT LEFT DOWN)) (SAW RIGHT DOWN) (MADE (PREDICT)) (CRIT
ERION WEIGHT) (WEIGHT BIG LEFT) (DISTANCE BIG RIGHT) (WEIGHT MORE LEFT) (DISTANC
E MORE RIGHT))

# CYCLE 5 END #
```

図 3 - (4)

```
# CYCLE 6 START #

: WORKING MEMORY AT START OF CYCLE 6 :
((RESULT WRONG) (DID (EXPECT LEFT DOWN)) (SAW RIGHT DOWN) (MADE (PREDICT)) (CRIT
ERION WEIGHT) (WEIGHT BIG LEFT) (DISTANCE BIG RIGHT) (WEIGHT MORE LEFT) (DISTANC
E MORE RIGHT))

= CONFLICT SET AFTER MATCHING =
((1 5) SW2)

= CONFLICT SET AFTER SPECIAL CASE ORDER =
((1 5) SW2)

(19) = SELECTED RULE AFTER WM ORDER =
     SW2

+++ MATCHINGFACT OF (RESULT WRONG) +++
(RESULT WRONG)

+++ MATCHINGFACT OF (CRITERION WEIGHT) +++
(CRITERION WEIGHT)

: WORKING MEMORY AFTER AUTOMATIC REHEARSAL :
((RESULT WRONG) (CRITERION WEIGHT) (DID (EXPECT LEFT DOWN)) (SAW RIGHT DOWN) (MA
DE (PREDICT)) (WEIGHT BIG LEFT) (DISTANCE BIG RIGHT) (WEIGHT MORE LEFT) (DISTANC
E MORE RIGHT))

: WORKING MEMORY AFTER FIRING :
((OLD (RESULT WRONG)) (CRITERION DISTANCE) (DID (EXPECT LEFT DOWN)) (SAW RIGHT D
OWN) (MADE (PREDICT)) (WEIGHT BIG LEFT) (DISTANCE BIG RIGHT) (WEIGHT MORE LEFT)
(DISTANCE MORE RIGHT))

: WORKING MEMORY AT END OF CYCLE 6 :
((OLD (RESULT WRONG)) (CRITERION DISTANCE) (DID (EXPECT LEFT DOWN)) (SAW RIGHT D
OWN) (MADE (PREDICT)) (WEIGHT BIG LEFT) (DISTANCE BIG RIGHT) (WEIGHT MORE LEFT)
(DISTANCE MORE RIGHT))

# CYCLE 6 END #

(20) :: WORKING MEMORY AT END OF TRIAL 1 ::
     ((CRITERION DISTANCE))

## TRIAL 1 END ##

(21) GO TO NEXT PROBLEM ?
     NO

### THE PROCESS END ###

NIL
```

図 3 - (5)