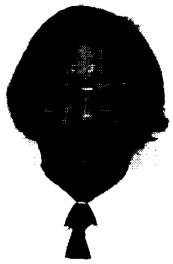
設立10周年記念特集号　随　想

情報処理の個別化・地球規模化時代における知識ベースシステムの進路



元田　浩*
Hiroshi Motoda

1. はじめに

「知識ベースシステム」研究会は「人工知能基礎論」研究会と「ヒューマンインタフェースと認知モデル」研究会とともに1987年6月に発足した．最初は「人工知能ツールと知識システム」研究会という名称であったが，その名前が使われたのは第1回目の研究会資料だけで，第2回目以降は「知識ベースシステム研究会」に改名され現在に至っている．この8年半の間に33回(1996年1月時点)の研究会が開催され，発表論文数も332件に及ぶ．この間，上野晴樹，溝口理一郎，小山照夫，筆者の4名が主査としてそれぞれ2年ずつ担当してきた．知識ベースシステム研究会はこの10年間の日本での人工知能研究の浮沈みを忠実に反映してきたといえる．発足当初のエキスパートシステムに対する期待を込めた熱気にあふれる発表と議論を思い出す．その後，しだいに個別の事例の経験などの発表から，根幹的な要素技術，既存技術との融合，知識ベース構築技術へと話題が移り，最近では知識の共有・再利用，機械学習との接点技術に力点が移動している．本稿では限られた紙面のなかで，知識ベースシステム関連研究の現状，問題点，近い将来に関する筆者個人の見解を述べてみたい．

2. 第一世代の知識ベースシステム

第一世代の知識ベースシステムでは種々の知識（タスク実行，領域特性，動作原理など）を明示的に区別することはせず，ひとまとめにして単一推論機構で処理していた．無数の個別の知識を組み合わせた複雑な推論連鎖から知的な振舞いが生成されるに違いないとの期待がそうさせたわけであるが，今では誰もこの考えを支持しない．結果は期待とは逆で，知識獲得，説明，頑強，保守，再利用などいずれの側面をとっても問題の性質を悪くしている．今では，中身が理解でき，かつ保守が容易なシステムを作るには，知識は明示的にモデル化しなければならないことを誰でも承知している．

問題解決プロセスにはいくつもの異なった側面があり，そのおのおので違ったモデル化が必要となり，これをどう処理するかが性能や効率を決める．どのような知識が必要か（知識レベル）ということと，それをどう実装するか（記号レベル）ということは，明確に区別しなければならない．このような反省から生まれてきたのが第二世代の知識ベースシステムと呼ばれるものであり，複数のモデル，複数の推論手法，知識レベルの設計などを特徴としている．

3. 第二世代の知識ベースシステム

(1) 複数モデル，推論手法の使い分け

一般に複数の知識源を使い分ければ，生産性も理解度も向上する．知識の塊が扱いやすい量に分割され，構造化が促進される．個別の問題にはそれに適した知識表現があり，各タスクを記述するのに必要十分なものを同定することが重要となる．表現が違えば違う推論手法が強調される．空間，時間，因果性，階層化の概念などはこのような視点で議論されて，相応の成果が出た．しかし，どのように知識を分割するかは問題であり，知識から表現へのマッピングにも自明な正解があるわけではない．異種の表現間の推論結果の整合性や伝播も問題となる．理論に基づくモデル（深い知識）を使った推論とそれとは独立に得られた経験的知識（浅い知識）を使った推論を組み合わせた診断システムはその典型例である．ニューラルネットや事例に基づく推論手法を組み合わせたものも多い．

＊　大阪大学産業科学研究所教授

（2） 知識レベルの設計

知識レベルという概念はニューウェルが最初に提案したものである．知的エージェントは合理性の原理に基づき行動するので，目標達成のために必要であると推論された行為からエージェントの挙動が説明・予測できる．つまり，知識レベルは挙動の「なぜ」の部分を説明する．ニューウェルが主張した知識レベルには構造の存在が明示されていなかったが，現在，知識ベースシステムで用いられている知識レベルの記述では構造化が重要な役割を果たしている．知識レベルと実装レベルを区別し，知識レベルを適切な抽象度で記述すれば規範的な設計，知識獲得支援，構成要素の再利用，より抽象度の高い説明などの点で効果が上がると考えるのは自然である．端的には，知識レベルでは専門家の挙動を知識でモデル化し，実装レベルでは記号と表現で相互作用をモデル化する．両者はともに観察される挙動をモデル化するという意味で整合性がとれている．

種々の知識レベルのアプローチに共通する点は，いずれも，①解決すべき問題の目標やそれを達成するための副目標を記述したいわゆるタスク，②これらの目標を達成するための方法，③その手法が必要とする領域知識（対象世界に関する記述）を基盤パラダイムとしていることである．そして，実装レベルでは性能向上に注力すればよい．その際，記号レベルの挙動を知識レベルの言葉で解釈できるような仕組みを準備し，推論結果の説明，知識の検証や獲得を知識レベルで実行できなければ意味が半減する．したがって，両レベルで構造が同相であることが望ましい．これも「言うは易く行うは難しい」．よく用いられる手法は，ビルディングブロック（知識レベルで記述された意味のある塊に対応した実装レベルの部品）を準備しておき，これを組み合わせてシステムを構築するのである．このようなモジュール化は実装面で優位であるが，ビルディングブロックには手法（解法）が固定されており，それ自身の修正（個別の問題に適合させるために必要になることがある）が難しい．タスク，解法を入れ子にし，自由度を増加するなどの改良が続けられている．野心的な試みとしては，知識レベルのタスク記述から直接実行可能な実装レベルのプログラムを生成するものもある．これらは枠組みは汎用であるが，現状では特定のタスクに限定した実験の成功が報告されているだけである．

（3） いまだに解決しない知識獲得問題

知識獲得のボトルネックは，知識ベースシステムが世の中に出てからつねに実用化への障害として問題にされてきた．知識レベルのアプローチは確かに知識獲得に対する見方を変えた．がむしゃらに専門家から知識を抽出するのではなく，専門家の問題解決過程をモデル化することが知識獲得そのものであるという考えである．これには二つのフェーズがあり，モデルそのものを作ることと，そのモデルに実体（領域知識）を入れることである．モデルは無から作られることはまずなく，新しいモデルはつねに汎用モデルの精緻化か既存のモデルの修正で作られるといっても過言ではない．モデルは構造であり，構造体としての部品には明確な役割がある．したがって，これを利用すればどの部分にはどのような種類の知識が必要かが明確になり，これをもとに知識獲得をガイドできるというのが知識レベルアプローチのうたい文句の一つである．

しかし，現実には適切なモデルを正しく選択する，あるいは精緻化によって作り上げるのは容易ではない．解決したい問題の特徴とモデルの特性との関係は往々にして非明示的である．それには，次項で述べる記述単位としての語彙の問題ともからむが，知識獲得のパラドックスの問題をすべてここに集約させているからである．「いかにして」に関する知識は，一般には暗黙知が関連する．「こと」に関する知識は意識できている知識である．心理学者は，暗黙知やコンパイルされた知識を手続き的，意識下にある知識を宣言的という．専門家の専門家たるゆえんは，豊富な前者の知識にある．皮肉なことに，経験を積み熟練すればするほど，宣言的な知識は手続き的な知識に移行し，「知っている」という自覚が失われる．したがって，一番知りたい知識は一番獲得しにくいというパラドックスが生じる．

言語や世界に関し我々が持っているメンタルモデルにより我々の思考が制限を受けるように，計算機で使用する言語の知識表現能力とそれをどう使いこなすかで知識ベースシステムの性能が決まる．専門家のメンタルモデルは，たぶん，最初はバラバラで暗黙的なものであろう．モデル化の際には，もやもやをはっきりさせ，ギャップを埋め，暗黙知を明確に記述しなければならない．知識獲得をガイドするのにモデルが有効であるが，そのモデルを獲得するのも知識獲得である．このように考えると，ここまで整理してしまえばできるのは当然だということになる．知識ベースシステム（ここでの議論にはエキスパートシステムのほうが語感が適切）に期待されたのは，悪構造でアルゴリズムが存在しない問題に対して有効に働く万能薬ということであったと思うが，結局のところ，細部まで透明なものを作らなければだめであるというのが結論で，それをいかにシステマティックに実

行するかの方法を模索していることになる．問題が整理された結果，知識獲得はモデル化のなかに埋め込まれてしまった．総論賛成各論問題ありというのが実体であろう．

（4）「言うは易く行うは難しい」知識の共有・再利用

知識レベルの設計を別の言葉でいえば，自分がどのような知識を獲得すべきかを知っているシステムの実現を目指した系統的な方法ということであり，このこと自体は重要な見識である．知識ベースシステム構築のための枠組みとしてその良さが認知されても，実際にシステムを作り上げるには知識レベルでタスクを記述するために必要な共通の語彙，各タスクの実行に必要な手法（解法）が要求する領域知識を記述するための共通の語彙に関する合意が前提となる．それが実現すれば，前項で触れた知識獲得のパラドックスもおおいに緩和される．専門家は与えられた語彙で語るように努力するようになる．しかし，現状ではそれがないため，知識の共有や再利用が非常に限定された範囲でしか実行できない．

もちろん，知識レベルという言葉がはやり出す以前から類似の考えは存在しており（Chandrasekaran の Generic Task や McDermott の Half Weak Method など），よく使われるタスク構造（ヒューリスティック分類，データ抽象化，カバー・差別，生成・検査，提案・修正など）はモジュール化されている．これらは一般的に（再）利用可能な工学的なノウハウと考えてよい．国際的に合意の得られる語彙の統一は難関であり，それに向けたオントロジーの研究が鋭意実施されているが，まだ見通しは定かではない．電子化辞書やひところ叫ばれた大規模知識ベースの構築が困難な原因もここにある．現在オントロジーに関しては，知識レベルの設計の流れと整合し，タスク（問題の性質）と領域（問題の対象）をできるだけ切り離して，それぞれのオントロジーを設計している．ここで，領域オントロジーをそれで記述されたものが何にどう使われるかを知らないで事前に記述することは，一般に可能かという素朴な疑問が生ずる．実体は，例えば領域オントロジーに部分全体の関係などを導入した瞬間に，その関係が十分一般的でいろいろなタスクに対する領域記述のプリミティブとして使えると想定しており，まったく独立ではあり得ない．

（5） 説明機能の向上

問題が難しければ難しいほど，わかりやすい説明が望まれる．筆者の経験でもアルゴリズムが理解でき，それをコード化したプログラムの各ステップで実行していることの意味が理解できても，問題の規模が大きくなればプログラムの出力する解が正しいかどうかはすぐにはわからない．理解できたと思う場合には，各自の理解能力に収まる範囲の枝葉末節を捨てた抽象化が行われている．計算機の推論ステップを細かく見せることはけっして良い説明ではない．

知識レベルのモデリングの利点は，このモデルの抽象度が人間の認知のレベルに合っているので，実際のプログラムの動きをこのレベルに逆にマップして専門家との間で合意のとれた語彙（オントロジー）で説明を構成できることである．そのためには，前にも述べたが，知識レベルと実行レベルでの対応がとれている必要がある．もう一つの考え方は，プログラムの実行と説明とを切り離すことである．つまり，実行は効率を主体に，説明は理解を主体に考えればよいとする，説明再構築法とでもいうものである．典型的な説明のパターンを準備しておき，結果をこれにマップし，説明を組み立てる．説明のための別の知識ベースが必要となるが，本来の問題解決プロセスとは別に，結果を正当化するもっともらしい筋道を組み立てることができる．マッピングの程度を調節すれば，問題解決プロセスに近い説明を生成することも，その逆も可能となる．説明用の知識ベースと問題解決用の知識ベースを分離することにより柔軟性は向上する．理解度が向上するが，逆に間違った説明を生成する恐れもある．この方向に沿った研究も散見されるが，まだ理想とはほど遠い．説明機能は推論機能に対する付足しと考えるのは間違いで，それ自体複雑な問題解決行為であり，そのための特別のアーキテクチャが要求される．

4．注目されている諸技術

（1） ファジィ推論とニューラルネット

ファジィ推論とニューラルネットは多くの知識ベースシステムで使われている人気のある手法である．両者ともそれ単独で使われることもあるが，全体の問題解決プロセスの一部を担当することも多い．これらの技術が使われるタスクはかなり明確である．ファジィ推論は制御によく使われる．少数の規則で人間の感覚に合った制御を実現できる点と，多入力・多出力が扱える点が魅力的である．簡単に実現できるので回路に組み込み，家電品などにも実装されている．このため，一般の人には人工知能＝ファジィ推論と誤解されたふしもある．ニューラ

ルネットは数値データを必要とするタスクによく使われる．特にパターン認識で威力を発揮する．学習したネットの構造が理解できないという批判もあるが，最近ではニューラルネットからルールを抽出する方法も研究されている．人間の脳と違い，可塑性が過去の学習を忘却するという問題を抱えており，インクリメンタルな学習のための理論強化が必要である．

（2） 事例推論

事例推論は実用的（ある意味で原始的）な問題解決方法である．事例を帰納によって一般化することをせずそのまま保存しておき，必要になって初めて一番近いものを取り出し，それに手を加えるという解決法はきわめて実践的である．事例そのものの存在が大きな意味を持つ設計や法律解釈などの分野で威力を発揮する．最大の特徴は解に至る複雑さが問題（解法）そのものの複雑さではなく，一番近い事例からの解の距離に依存する点であり，知っていることの強みを最大限に発揮している．事例の特徴のインデキシングの体系化が困難である点はよく指摘される．また，解の修正のためには別の知識が必要となる．事例が蓄積するにつれて各事例がカバーする範囲に重なりが生じ，ここでもユーティリティ問題（知れば知るほど性能が悪くなる現象）が発生する．不要な事例を消去する必要があるが，領域理論の存在を前提とする説明に基づく学習とは違い，昔解けた問題が解けなくなるという性能劣化（単なる効率劣化ではない）に注意しなければならない．いかにして忘れるかということも重要な研究テーマである．

（3） 遺伝的アルゴリズム

遺伝的アルゴリズムは，適応進化や最適化の手段として，知識ベースシステムのなかに組み込まれる．したがって，応用は多種多様である．しかし，この方法がうまくいくかどうかはまだ経験的な側面が強い．新しいエンコーディング法と探索法が鋭意研究されている．プログラム生成やプラン生成におもしろい結果が出ている．

（4） 人工生命

知識ベースシステム（その前提となる人工知能はもちろんのこと）は明示的な知識の存在を前提にしている．知識レベルのアプローチを見てもわかるように，複雑なものでも構造化して単純なものに分解すれば全体が理解できるという還元論に基盤がある．つまり，ニュートン，デカルト以来の近代西洋科学の思想に立脚している．人工知能では，人間は知的で，その知的な人間の振舞いを人間が理解したうえで，計算機上で再現しようという立場をとる．これに対し，人工生命では，人間の考えることなんてたいしたことはない，したがって，事前に準備した明示的な知識は必要でない，との立場をとる．そして，無数のミクロな操作による相互作用によって知能は創発し，それが逆にミクロな挙動を制御すると期待する．近年，この考え方の限界と可能性を模索するためのシミュレーションが多数実施されている．地図なしロボットのナビゲーションや群生する鳥の生態把握などで良い結果が出ており，この方法の未来は明るいようにも見える．しかし，成功するのはまだ特定の種類の問題に限定されており，人工知能に置き代わるのはまだ遠い先のように思われる．しかし，ニュートン以来まだわずか300年しか経過しておらず，明示的な知識を前提とする現在主流の人工知能が万能であると信奉するのは危険である．

（5） サブサンプションアーキテクチャ

この考えも，計算による創発という点で人工生命と同じ基盤上にある．知識ベースシステムのコミュニティで正面きって取り上げられたことはないが，ロボット産業に大きなインパクトを与えるものと予想する．極度のモジュール性，低廉な製造コスト，ソフトウェアインプリメンテーションの容易さ，それに加えて変化する環境への適応能力の創発を考えると，この技術の将来は明るい．過去に大規模な設備投資をした大企業はまったく発想の違う技術には簡単に移行できないであろうから，小規模ベンチャーの活躍が予想される．

5. 欠けているユーザの視点

以上，ここ数年間の本研究会や関連学会ならびに論文誌などの発表からの技術動向に筆者の見解を交え，知識ベースシステムの現状を述べてきた．このような技術を深めていけば，その延長線上に我々が考える知識ベースシステムの理想像があるのであろうか？

近年の情報処理の個別化と地球規模化の同時進行を目の当りに見，そのなかで仕事をしていると，知識ベースシステムは，特殊な人々が特殊な仕事をするためにのみあるのではなく，今後は一般ユーザのさまざまな要求に答えられる知的な支援システムとしての役割が強く要求されるものと確信する．一般ユーザは，知識ベースシステムのおかげで，余裕やゆとりが増え，無味乾燥な単純秘書業務から解放され，仕事は加速され，質も向上する

と期待する．現に，オフィスオートメーションはこのような効果を期待して導入されたものである．しかし，実際はどうであろうか？　余裕やゆとりは減少し，仕事はますます多忙にかつ難しくなり，朝から晩までキーボードの奴隷と化している．ペーパレスといいながら，むだな紙を多数出力している．

なぜこのようなことになるのであろうか？　原因は単純で，知識ベースシステムはそれを使う個別のユーザのことを何も知らないからである．一人一人のユーザは，似たものに興味を持っていても，それは微妙に違うものである．これからは，領域，タスクのモデル化に加えて，個別ユーザのモデル化が重要になってくる．ユーザ自身，自分の興味や考えを徐々に変えていく．したがって，ユーザに負担をかけることなく，このような環境変化に追従する仕組みが必要となる．機械学習やインタフェースの研究と切り離せなくなる．

人工知能では普通の人ができないことを代行するほうが，普通の人が当り前にできることを代行させるよりはるかにやさしい，とよくいわれる．画像理解や自然言語理解などは，後者の典型例である．知識ベースシステムとは別の領域（現に本学会でも言語・音声理解と対話処理研究会が別にある）のものとして扱われたものが，結局，知識ベースシステムに要求されることになる．

6. おわりに

知っていることの強みは，未来永劫変わらないであろう．知識ベースシステムは，この事実を最大限に利用しなければならない．性能を発揮するために個別の問題に対し「狭く深く」と知識を追求してきた（せざるを得なかったというべきか）知識ベースシステムも，時代の要求に応えるためには「広くきめ細かく」が要請されるようになってきた．汎用知能への回帰が必然的な流れなのであろうか？　それは，現在の知識ベースシステム研究の延長線上にあるのであろうか？　それとも，まったく別のアーキテクチャが要求されるのであろうか？　今こそ，基礎研究に投資すべき時期である．