

特集 「類似性に基づく推論」

創造性と類似性

Creativity and Similarity

折原 良平
Ryohei Orihara

(株) 東芝研究開発センター知識メディアラボラトリー
Knowledge Media Lab., Corporate R & D Center, Toshiba Corporation.
ryohei.orihara@toshiba.co.jp

Keywords: metaphor, creativity, analogy, perception.

1. はじめに

創造性は人間の知性の発露の最たるものである。それゆえ、創造性のメカニズム解明は哲学、生物学から計算機科学に至るまでたくさんの学問分野において興味を尽きないテーマであり、最近では「発見科学」プロジェクト [有川 00] もこの系譜に連なるものである。創造性は、どのような心の働きによってもたらされるのか。これを分析するために、伊東は過去の重要なアイデアの発見をもたらした発想の形式を以下のように分類した [伊東 83]。

- 類推によるもの：ある分野（ターゲット領域）における未知のアイデアを、その分野と別の分野（ベース領域）の類似性に基づき、ベース領域の要素をターゲット領域に転写することで獲得する。
- 普遍化によるもの：与えられた既知の複数の理論をある観点から統一的に捉えることで、より一般的な理論を構築する。
- 極限化によるもの：経験的事実の理想的極限を考えることにより、新たな法則を発見する。
- システム化によるもの：数多くの事実に対し独自の観点からそれらの間の関係性を見抜く。

伊東は、この中で類推が最も重要であると述べている。特に歴史的な数量データが示されているわけではないが、ダーウィンの自然選択説が、人工的な変種をつくり出す品種改良からの類推で得られたことや、ラザフォードの原子モデルと太陽系 [Falkenhainer 86] など、歴史上有名な発見を支えたとされる類推は枚挙にいとまがなく、発想のエンジンとして類推が重要であるという説には直感的にうなずかされる。

ところが、最近になって、歴史上有名な発見に寄与したとされる類推が、実は発想段階では用いられておらず、新しいコンセプトを他人に説明するために使われていた可能性が高いという報告が現れた [Rhodes 88]。本解説では、はじめに、実データに基づいて科学者の思考を分析することによりこれらの報告を否定し、類推が発想にとって重要であることを証明した植田の研究を紹介する [植田 00]。植田は、同時に、類推の形式について興味深

い分類を行っており、その中で、「非常に大きな理論変化をもたらす可能性がある」と指摘しているタイプの類推は、Indurkha がメタファ理解の枠組みの中で注目してきた認知モデル [Indurkha 92] によって説明できるものである。3 章ではこのモデルを紹介する。4 章では、Indurkha のモデルと方向性を共有する二つの計算機シミュレーションモデルを紹介し、それらの特徴と、「伝統的」類推システムやメタファ理解システムとの違いを説明する。5 章では、こうしたモデルの工学的応用として発想支援システムの可能性を模索する。最後に、6 章で本解説を総括する。

2. 創造的類推

発想のエンジンとして類推が有効であるという説は直感的に受け入れられやすいが、その根拠となると歴史的文献や科学者の逸話に基づいていることが多く、そこで引用されている類推の例が実際に重要なアイデアを導いたのかどうか定かでない。そこで、認知心理学では、複雑ではあるが比較的短時間で解ける実験課題を被験者が解くプロセスを分析し、科学的発見を導く認知プロセスを明らかにしようとしてきた (*in vitro* アプローチ)。しかし、(1) 認知心理実験で用いられる課題が科学者の扱うものとかい離していること、(2) 被験者が科学者ではないこと、(3) 科学的発見は非常に長い時間をかけて到達されるものも多いこと、から、*in vitro* アプローチで観察されるプロセスは科学者による発見と同じとは言えないという批判があった。植田は、現場の科学者の活動データの分析 (*in vivo* アプローチ) を用いて、類推が現実に科学者の発見に寄与していることを示した [植田 00]。

植田は、日本の自然科学および工学の各分野の一線級研究者 22 名に対し、彼らが現在関わっている、あるいは直前まで関わっていたプロジェクトに関してインタビューを行い、その内容を研究メモや共同研究者からのインタビューにより検証しながらデータを収集した。その結果、22 人から収集した 42 事例のうち、約 45 % の 19 事例で類推が用いられていることがわかり、その使用頻度が高いことが証明された。

次に、植田はこの 19 事例を「どんな類似性が用いられたか」、「類推によって何が転写されたか」の 2 軸から分類した。前者は目的指向型（ターゲットのゴールと類似な事実を有するベースを探す）、既存カテゴリに基づく（ゴールではないが、ターゲットのカテゴリと類似なカテゴリをもつベースを探す）、新カテゴリ形成に基づく（既存カテゴリに基づいて検索されたベースとターゲットの対応の不完全性を補うため、既存カテゴリが拡張される）の 3 分類を与える。この中で、植田は「非常に大きな理論変化をもたらす可能性がある」として第 3 のタイプに注目した。このタイプの例は「球状星団の特異な挙動を説明する理論の構築」である。この理論を構築した研究者は、「自己重力系」という類似性に基づいて恒星と球状星団の類似性を認めることで理論を構築しようとしていたが、従来の理論のもとではその類似性は不完全であった。この研究者は、偶然知った非平衡熱力学と線形安定化理論を用いて既存の理論を拡張することにより、類似性を新たなカテゴリである「自己重力-熱力学系」に拡張し、より完全な写像と転写を可能とした。カテゴリの拡張は、理論の多くの概念の修正を導き、重要な知見をもたらした。カテゴリの拡張は帰納に基づくことから、植田はこの例をもって「類推のみによって知識のゲシュタルト的再構成が起こるわけではない」としているが、筆者は逆に、カテゴリの拡張はより大きな類推プロセスの一部であり、まさに類推によって知識のゲシュタルト的再構成が起こった例であると考えたい。

3. 創造的類似性の検出モデル

Indurkha はメタファの認知的・数理的モデルを研究してきており [Indurkha 87, Indurkha 92], 植田の新カテゴリ形成に基づく類推と深く関連するモデルを提案している。

Indurkha はメタファを *Similarity-Based Metaphor* と *Similarity-Creating Metaphor* に分類した。前者は既存の概念の類似性に基づくもので、後者はそうではなく、メタファを理解する過程で類似性が認識されるようなものである。例えば、次の詩を考えよう。

<p>The fog comes on little cat feet.</p>	<p>It sits looking over harbor and city on silent haunches and then moves on.</p>
--	---

霧と猫の間の類似性は事前には存在しないが、詩を理解するにあたって読者の脳裏につくられた両者の表象の間には確かに類似性がある。これが *Similarity-Creating Metaphor* である。

2 種類のメタファを説明するため、Indurkha は 2 階層の概念化のモデルを導入した。これは、対象が抽象度の異なる二つの層の両方において表現され、それら表現の相互作用でメタファが説明できるというものである。

抽象度の低い層を *realm*, 高い層を *concept network* と呼ぶ。すなわち、情報の受容は *realm* → *concept network* の向きに進む。直感的には、*realm* は知覚された刺激の層であり、*concept network* は対象に関する認知主体の背景知識である。

これら 2 層の間には 2 種類の重要な相互作用がある。*accomodation* は、*realm* に基づく *concept network* の変化であり、*projection* は *concept network* に基づく *realm* の変化である。直感的に言えば、*projection* は、

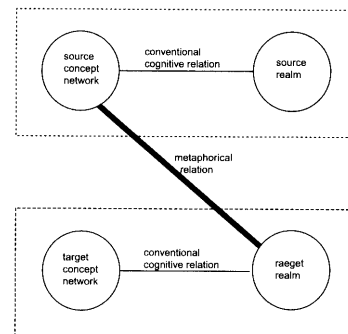


図 1 Interaction between Concept Network and Realm (adapted from [Indurkha 92])

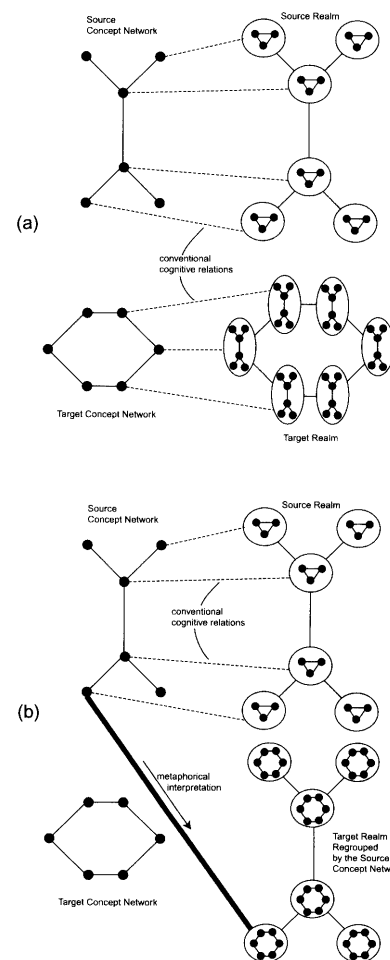


図 2 Similarity-Creating Metaphor (adapted from [Indurkha 92])

主体が未知のものを既知のもの（に近いもの）として捉えようとするプロセスにあたる（図1）。

このモデルでは、Similarity-Creating Metaphorは、ベース対象の concept network からターゲット対象の realm への projection が、ターゲット対象の concept network と無関係に起こる場合として説明できる（図2）。図中、(a)はメタファ理解前を表し、ベースとターゲットにおいてそれぞれ独立に、通常の認知的関係に基づいて、concept network と realm が結ばれている。concept network どうしあるいは realm どうしに類似性は見られない。(b)はメタファ理解後のようすで、ターゲット realm はベース concept network の構造を反映するように再構成され、その結果ベース realm とターゲット realm の間に類似性が発生している。

Indurkha のモデルは、なぜ最初に特定のベースとターゲットが選ばれるのかを説明しない。上記の詩の例で、霧と猫がベースとターゲットとして意識されるのは、詩人によってそれらが併置されているからである。植田の新カテゴリ形成に基づく類推では、既存カテゴリに基づいて検索されたベースとターゲットとの対応が、新カテゴリによって補強されるとしていた。その意味で、植田の新カテゴリ形成に基づく類推は、科学者の先入観あるいは固執によってベースとターゲットが選ばれる場合の Similarity-Creating Metaphor として説明できる。

4. 創造的類似性検出の実現

Indurkha は前章で述べたモデルが現状の人工知能技術によって実現可能であると議論している [Indurkha 92]。彼の言葉を証明するように、Similarity-Creating Metaphor と類似の機能を実現する計算機上のモデルが開発されている。本章では、二つの研究を紹介する。

4.1 Copycat

サッチャー女史が首相であった頃の英国で「英国のファーストレディは誰か？」という質問（答の一例はデニス・サッチャー氏）に答えるためには何が必要だろうか。英国と米国における政治家や王室を結ぶ関係を、あるときは制約と考え、あるときは緩和しながら妥当な対応をつくり上げる柔軟なメカニズムが要求されるだろう。このような問題意識のもとで Copycat [Mitchell 93] は設計された。

Copycat は、類推システムであるが、問題解決を目的としたものではなく、知覚のモデルたるアナロジー生成のモデルをつくらうとしている。扱っている問題は知能テストに出てくるような文字列置換問題、例えば“abc → abd; ijk →?”（「abc は abd に書き換えられるとしましょう; 同じことを ijk にしてごらん下さい」）である。この問題には明確な答はない。Copycat は正しい答を出すのが目的ではなく、こうした問題を人間がどう捉

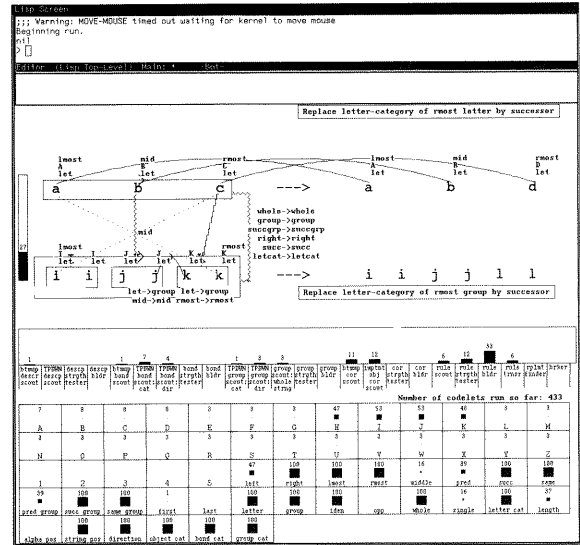


図3 A Test Run of Copycat

えるかをモデル化しようとしているのである。そこで、人間がそうであるように、同じ問題に対して実行ごとに異なった答を返すことがあるように設計されている。例えば、上記の問題に対しては、ijl のほか、ijd や ijk が返されることもある。図3は、“abc → abd; iijjkk →?” に対する Copycat の実行画面の一例である。

Copycat のアーキテクチャは、Slipnet, Workspace, Coderack, 温度の4部分からなる。Slipnet は長期記憶に相当し、アルファベットや算術に関する知識を意味ネットの形で保持する。Workspace は短期記憶に相当し、与えられた文字列に対する表象 (eg. a から始まる昇順グループ) がつくられる場所である。表象は、簡単な特徴や構造を検出する codelet と呼ばれる知覚エージェントが多数起動されることによってつくられる。その起動を制御するのが Coderack である。温度はシステムの振舞いのランダムネスを制御する。

“x → y; z →?” という問題が与えられると、Copycat は codelet を起動し、各文字列の解釈を試みる。例えば、文字列 abc 中の a という文字は A というアルファベットのインスタンスでもあるし、最も左の文字でもある。このような解釈は description と呼ばれ、ボトムアップ（何らかの理由で注目している文字に description を割り当てる）およびトップダウン（何らかの理由で注目している description に適合する文字を探す）に文字に割り当てられていく。隣り合った文字の description の集合に対しては同様に bond と呼ばれる解釈が割り当てられていく。例えば昇順 bond, 同種 bond などで、この処理も codelet の起動による。ここでも、ボトムアップおよびトップダウンの処理がある。同じタイプの bond でつながれている文字は group を形成することができ、これも codelet によって実現される。さらに、文字列 X と文字列 Z の間で文字や group の description どうしの間に対応を発見する codelet や、文字列 X から文字列 Y への

書換えのルールを提案する codelet が存在し、最終的に X と Z の対応に基づいて書換えルールから類推して得られたルールを Z に適用することにより答を出力する。

どのトップダウン codelet が起動されるか、および Workspace 中でどの記述が注目されるかは、Slipnet 中の概念の活性に依存して確率的に決定される。活性はリンクを通じ Slipnet 内を伝搬する。これにより、ある概念と関連する概念に関して解釈を続けることが可能になっている。確率的選択のランダムネスは温度によって制御されており、最初は高く、bond や group の形成に従って下がるように設計されている。

文字列を解釈して対応と書換えルールを発見するという一連の流れを細かく分解し、codelet の実行列によって実現しているため、Copycat は複数の可能性を同時に探ることができる。これを *parallel terraced scan* と呼ぶ。この実現に欠かせないのが、つくられた構造を確率的に破壊する *breaker codelet* である。

こうした構成により、Copycat は “abc → abd; mrrjjj → ?” に対して mrrjjjj のように「創造的」な答を返す能力ももっている。

Similarity-Creating Metaphor の観点から Copycat を見てみると、文字列の解釈である *description* が動的につくられるという点で、Copycat は既知の類似性に基づかない類推システムのうまい例になっているといえる。

4.2 Sapper

Sapper [Veale 94, Veale 97] は、意味ネットワークに基づくメタファ理解のモデルである。意味ネットワークは記憶のモデルとなっており、概念の活性がリンクを通じて伝搬するという局所表現のネットワークになっている。Veale は、メタファが字義どおりの文に還元できないという知見に基づき、この記憶のモデルにおけるメタファ理解においては、活性伝搬とは全く異なるメカニズムが働いているべきであると考えた。そこで、次のような 2 段階のメタファ理解のモデルを考案した。

- (1) **dormant bridge** の生成：メタファを担う記憶構造の候補を、シンボリックなシステムが生成する。
- (2) 活性による **awakening**：実際にどの候補を用いてメタファが理解されるかは、活性伝搬で決まる。**dormant bridge** の生成は以下のルールに基づく。
 - **Triangulation Rule**: 概念 C_i と C_j がともに C_k との間と同じタイプのリンクをもっているとき、 C_i と C_j の間に **bridge** を生成する。
 - **Squaring Rule**: C_j と C_k の間に **bridge** があり、さらに C_i と C_j , C_i と C_k がそれぞれ同じタイプのリンクでつながれているとき、 C_i と C_j の間に **bridge** を生成する。

図 4 は **bridge** 生成の例である。まず、**Triangulation Rule** により、「中央」という属性を持つ「脳」と「指令室」の間に **bridge** が生成される。次に、「脳」と「脳外

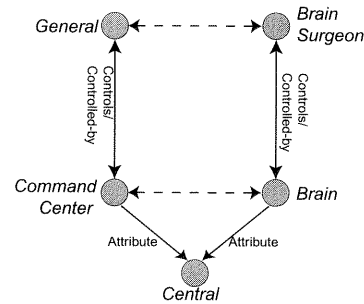


図 4 Sapper's Heuristics (adapted from [Veale 94])

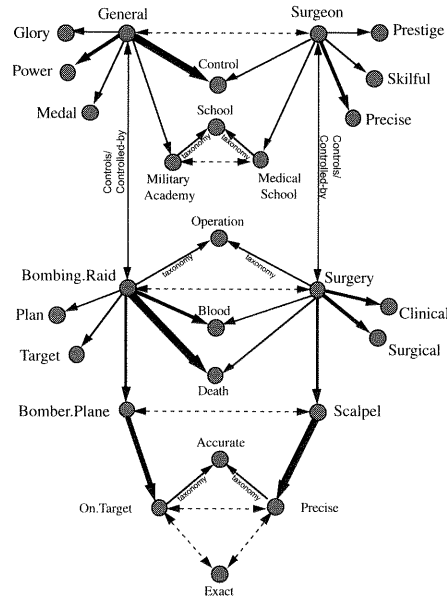


図 5 “The General is Surgeon” (adapted from [Veale 94])

科医]、「指令室」と「将軍」はともに「制御する・される」という関係で結ばれているので、**Squaring Rule** により、「脳外科医」と「将軍」の間に **bridge** が生成される。

図 5 は、この要領でつくられたネットワークの例である。メタファを理解するための対応関係は、**bridge** によって拡張されたネットワークに対し、ベースとターゲットの概念ノードを出発点に活性を伝搬させることで行われる。ある **bridge** で二つの活性が会ったとき、メタファが理解されたと考え、活性伝搬経路上の概念を用いて対応を出力する。

Similarity-Creating Metaphor の観点から Sapper を見てみると、対象のどの部分が注目されるべきかがメタファが理解されて（活性が会って）はじめて判明するというモデルになっている。ただ、活性伝搬の前に **dormant bridge** をすべてつくっておくというプロセスが不自然に感じられるのは否めない。これについては次節で議論する。

4.3 比較および伝統的システムとの違い

ここでは、Copycat と Sapper の比較および著名な類推システムとの違いについて議論する。

§ 1 構造の生成方式

Copycat における構造を生成する codelet に対応するのは、Sapper では生成された dormant bridge である。すなわち、codelet の集合は可能なすべての構造を表しており、それは Sapper においては dormant bridge 生成ヒューリスティックではなく、dormant bridge そのものなのである。Copycat においては、そのときの状況に応じて codelet が実行され、可能性のうち一部の構造がつくられる。これに Sapper で対応するのは、「使われた (burnt-in)」bridge である。このように、この点に関して論理的には両者は同等だが、Sapper におけるヒューリスティックの適用プロセスがしらみつぶしである点が認知モデルとして不自然に感じられる。実際には、Sapper におけるヒューリスティックの適用は、Copycat においては codelet や構造に関する概念を表す Slipnet の設計にあたり、Copycat においてさえダイナミックなプロセスとしてはモデル化できていないのだが、Copycat がそれを感じさせないのは、文字列という Microworld を対象としており、Microworld の全知識を記述するという行為の中に、可能な構造をすべて列挙することが隠されているからであろう。Sapper はあらかじめ決められていないさまざまな領域を抜おうとするため、領域非依存の基本部分（活性伝搬とヒューリスティック）の上に領域知識が乗るというアーキテクチャになっている。Copycat では基礎部分に領域知識が入っている。うがった見方をすれば、領域非依存の認知モデルはあり得ないという哲学を反映しているとも取れる。それが正しいかどうかは別にして、Copycat の設計は巧みである。

§ 2 実行の制御

Copycat の parallel terraced scan に対し、Sapper は活性伝搬により、対応の並列探索を実現している。両者が大きく異なるのは、Copycat が非決定的であるのに対し、Sapper は決定的な点である。もちろん、非決定的なほうが認知モデルとして自然と思われる。しかし、Sapper に非決定性を導入するのは、活性伝搬のメカニズムを工夫することで、それほど困難なく実現可能であろう。

§ 3 伝統的システムとの比較

ここでは、過去の代表的な研究である Structure Mapping Engine (SME) [Falkenhainer 86], ACME [Holyoak 89] と比較して、Copycat や Sapper の特異点を説明する。まず、簡単に SME と ACME を復習する。

- SME は、構造写像理論 [Gentner 83] に基づく領域間の対応を発見するシステムである。構造写像理論は、述語論理式のネットワークで表現された知識の間でどのような写像がなされるべきかを決定するシステム性原理を与えている。それは、単純な属性に比べて、より高次の関係や、関係の関係などのほうが領域間で保存されやすいとするものである。

- ACME は、領域間での記号の対応の整合性の維持を、Hopfield ネットワークを用いた制約充足問題として解くシステムである。記号の対応がノードに割り振られ、両立する対応どうしは興奮性リンクで、良質しないものどうしは抑制性リンクで結ばれる。

Mitchell [Mitchell 93] は、Copycat と SME, ACME の詳細な比較を行っている。この中で、最大の違いは、SME/ACME がともに述語論理風の形式で対象が与えられることを仮定しているのに対し、Copycat が対象の表現を状況に応じて生成できることとしている。もちろん、与えられた表現にはなかった構造を付け加えることのできる Copycat や Sapper の能力は特筆されるが、SME, ACME で与えられる表現を Sapper における dormant bridge を含む表現と考え、その中からメタファを理解するために注目すべき対応を選択している問題と考えれば、前項で見たように、それほど大きな違いではない。

SME との最大の違いは、SME が領域依存のトップダウンな制御を行わないのに対し、Copycat ではトップダウン codelet として、Sapper では活性伝搬の形でそれが行われる点である。メタファ理解への interactionist 的立場 [Indurkha 92] からは、この違いは決定的である。

表現の動的生成を別にして、ACME は、すべての対応をしらみつぶし的に考慮する点が Copycat と異なり、また Sapper には類似する。また、ACME は pragmatic unit と呼ばれる、ネットワーク中で注目すべきノードを特別扱いするための機構をもっており、若干のトップダウンな制御を行う。しかし、pragmatic unit は事前のプログラミングの産物であり、Copycat や Sapper のようにトップダウンな圧力が実行中に創発するわけではない。

以上をまとめてみる。従来の類推システムがモデル化しているのは、類推の対応付けの段階のみであり、それ以前にあるはずの、状況を解釈し記述する部分が抜け落ちている。Copycat はこの問題をエレガントに解決し、必要な構造を必要に応じて生成していくモデルを実現した。一方、Sapper の解決は Copycat に比べると不器用であり、可能な付加的構造をすべて生成した後、その一部をダイナミックに選ぶという方式である。しかし、Copycat の成功は、メタファ課題のうち構造生成の可能性を列挙する部分を課題から切り離し、複数の課題に対して使用可能なシステムの一部 (Slipnet と codelet) としたことによっており、その結果領域知識をアーキテクチャの深部で使うことになり、領域依存性をもつこととなった。また、Sapper との見掛け上の違いは大きいが、本質的には両者とも可能な構造の候補からの選択である。

5. 発想支援への応用

科学的発見を支える認知プロセスが Similarity-Creating Metaphor の理解と深く関わっているとすれば、Copycat や Sapper のメカニズムを利用して高度な

発想支援システム [折原 93a, 折原 93b] を構成することが期待される。例えば、次のような機能を実現することができるだろう。

- メタファのベースとターゲットとしては考えにくい対象の対 (例えばランダムに選ばれたもの) に対し、対応を発見することにより、今までとは違った観点を獲得。あるいは、得られた新規なメタファに基づいて、キャッチフレーズや文学作品を作成する。
- よく知られた概念や製品に対応する、別の分野の仮想的な概念や製品がもつべき特性を推論する。Sapper を用いた商品企画を考えよう。企画中の商品 X の属する分野 A と、ほかのヒット商品 Y およびそれが属する分野 B のドメインモデルを構築し、分野 A のモデルには、この分野のすべての概念と dormant bridge で結合した仮想的な商品として X を記入しておく。Sapper が X と Y の対応を発見することで、分野 A 中で X がもつべき特性が明らかになる。

いずれの場合も、ドメインモデルの構築が困難であることが予想される。解決案としては、WWW ページのリンク関係に基づいてモデルを構築したり、テキストマイニング技術 [市村 00, 市村 01] によってドキュメントから概念とその関係を抽出することが考えられる。

従来も、類推に基づいてアイデアを提案する発想支援システムは存在した。知恵の泉* [折原 94] は、与えられた知識ベースを複数通りの方法で領域に分割することにより、一つの知識ベースから複数通りの類推結果を得ることができるシステムで、例えば製品のデータベースを与えれば、新製品の提案を行うことができる。実験により、知恵の泉の発想支援効果は、新規な領域分割の方法の発見に基づくことがわかっている。Copycat や Sapper との関連で述べると、領域分割の仕方を変えることは対象の表現を間接的に変化させることに対応しており、それが発想支援につながっていることは、対象の表現を生成することが Similarity-Creating Metaphor のメカニズムを実現していることに対応する。知恵の泉は完全にボトムアップなシステムで、状況に応じて領域分割の仕方を変えることはできないが、Copycat や Sapper のようなトップダウン圧力をうまく導入できれば、より大きな発想支援効果を得ることができるかもしれない。

6. おわりに

創造性と類似性に関して、創造性に貢献しているタイプの類推の同定、それを説明する認知モデル、さらにそれを計算機上を実現するシミュレーションモデルを紹介し、関連研究と比較するとともに、新しい発想支援システムの可能性について考察した。創造性は人間の人間た

るゆえんであり、IT 技術の最終的なゴールとしてその支援への期待は高い。多くの分野の成果が結集して、新たな展開を見せることに期待したい。

◇ 参考文献 ◇

- [有川 00] 有川節夫, 佐藤雅彦, 佐藤泰介, 丸岡章, 宮野悟, 金田康正: 発見科学の構想と展開, 人工知能学会誌, Vol. 15, No. 4, pp. 595-607 (2000)
- [Falkenhainer 86] Falkenhainer, B., et al.: The Structure-Mapping Engine, in *Proc. AAAI' 86*, pp. 272-277 (1986)
- [Gentner 83] Gentner, D.: Structure-Mapping: A Theoretical Framework for Analogy, *Cognitive Science*, Vol. 7, pp. 155-170 (1983)
- [Holyoak 89] Holyoak, K. and Thagard, P.: Analogical Mapping by Constraint Satisfaction, *Cognitive Science*, Vol. 13, No. 3, pp. 295-355 (1989)
- [市村 00] 市村由美 ほか: 営業日報を対象としたテキストマイニング-成功事例および機会損失情報の抽出-, 2000 年度人工知能学会大会論文集, pp. 532-534 (2000)
- [市村 01] 市村由美, 長谷川隆明, 渡部勇, 佐藤光弘: テキストマイニング-事例紹介, 人工知能学会誌, Vol. 16, No. 2, pp. 192-200 (2001)
- [Indurkha 87] Indurkha, B.: Approximate Semantic Transference: A Computational Theory of Metaphors and Analogies, *Cognitive Science*, Vol. 11, pp. 445-480 (1987)
- [Indurkha 92] Indurkha, B.: *Metaphor and Cognition*, Vol. 13 of *Studies in Cognitive Systems*, Kluwer Academic (1992)
- [伊東 83] 伊東俊太郎: 科学における創造性, 日本創造性学会 (編), 創造の理論と方法, pp. 73-84, 共立出版 (1983)
- [Mitchell 93] Mitchell, M.: *Analogy-Making as Perception: A Computer Model*, The MIT Press (1993)
- [折原 93a] 折原良平: 発散的思考支援ツールの研究開発動向, 人工知能学会誌, Vol. 8, No. 5 (1993)
- [折原 93b] 折原良平: 発想支援システムの動向, 情報処理, Vol. 34, No. 1, pp. 81-87 (1993)
- [折原 94] 折原良平: 発想支援システム「知恵の泉*」, 人工知能学会誌, Vol. 9, No. 2, pp. 248-257 (1994)
- [Rhodes 88] Rhodes, R.: *The Making of the Atomic Bomb, Touchstone* (1988)
- [植田 00] 植田一博: 科学者の類推による発見, 人工知能学会誌, Vol. 15, No. 4, pp. 608-617 (2000)
- [Veale 94] Veale, T. and Keane, M. T.: Metaphor and Memory: Symbolic and Connectionist Issues in Metaphor Comprehension, in *Proc. of the ECAI Workshop on Neural and Symbolic Integration* (1994)
- [Veale 97] Veale, T. and Keane, M. T.: The Competence of Sub-optimal Theories of Structure Mapping on Hard Analogies, in *Proc. of 15th IJCAI*, pp. 232-237 (1997)

2001年11月20日 受理

著者紹介



折原 良平 (正会員)

1986年筑波大学第三学群情報学類卒業。1988年同大学院工学研究科電子・情報工学専攻博士前期課程修了。同年、(株)東芝入社。現在、同社研究開発センター知識メディアラボラトリーに勤務。1993～95年にかけて University of Toronto 客員研究員。博士(工学)。発想支援技術、類推、機械学習、データ・テキストマイニングの研究に従事。情報処理学会、日本ソフトウェア科学会各会員。