診療ガイドラインによる診療内容確認に関する研究

診療ガイドラインの手順知識記述に関する研究

研究要旨　：　診療ガイドラインの手順内容を電子的に記述した場合に、材料である用語を基底層、手順の連なりや「AならばB」のような論理的な関係を手順層、そして得られた電子的記述を用いて診療などの場に利用する部分を応用層と呼ぶと、本研究では従来の記述法における反省点を踏まえて、特に手順層に関する検討を行った。記述法を改良することにより、手順に含まれる状態や行為、条件分岐で評価されるデータ、目的を同じくする一連の手順の集合である診療ブロックに対する記述が容易にできるようになり、可読性を高めることができた。またAならばBという条件判断について検討を行い、論理的な含意と実生活での「ならば」との乖離をはじめとして、可能性と必然の区別、因果関係と共起の区別などを意識した上で、判断ルールを記述することが重要であり、最終的に人間による判断を仰ぐための仕組みを導入する必要性について論じた。

A.　目的

診療ガイドラインを電子化する際には、知識記述の基本要素である語彙や概念を規定した基底層とこれらを組み合わせて手順を記述する手順層とに分けられる。特に手順層では個々の基本的な手順要素において、その目的と終了条件を明確に記述しなおすことを基本とし、さらに手順の分岐において可能性と必然とを区別することを通じて、手順層の記述を検討することを目的とする。以下でそれを詳細に述べる。

A.1.　診療ガイドラインの電子化

診療ガイドラインは文章（テキスト）で記述されている。テキストを人間が読み、内容を理解することによって、EBM に基づいた最適な診療が行われることを想定するものである。しかし臨床現場の多忙な医療従事者に対し、テキストで配布された診療ガイドラインを、それも多数の疾患に亘る複数のガイドラインを熟読し、それらの内容を正しく理解することを期待するだけでは、診療ガイドラインの普及が十分に成されるとは期待できないであろう。さらに診療ガイドラインは数年ごとに改訂される。過去に理解した記憶を最新の内容に更新することに困難を伴うことがあることは容易に予想される。知識更新の対象を各自の専門分野に絞れば容易であるとしても、診療全般に亘る知識の更新には、より効率的な方策が考えられるべきである。

最近の医療には診療システムが導入されており、例えば電子カルテやオーダリングシステムはすでに多くの医療施設に導入されている。検査

データや診察所見など患者に関するデータは、いずれこうした電子カルテの上に蓄積されることになろう。医療において投薬指示などの介入を行う際には、こうした電子的に保存された患者データを自動的に参照し、判断支援システムを介することにより、誤った指示を減少させうることが諸家の研究により示されている。

診療ガイドラインについても、EBMに基づいて推奨あるいは非推奨とされている医療的介入事項を、単にテキストとして配布するだけではなく、診療情報システムと連動させた判断支援システムを活用することによって、医療従事者を支援することが有用であると考えられている。そのためには、診療ガイドラインの内容を整理・簡略化し、フローチャートの様な形で判断を行う際の情報や条件を提示し、条件に合致した場合の処置などの手順を明確にすることが必要とされる。

患者の側に立っても、このような形式に診療ガイドラインが変換されたならば、専門用語で記載された膨大な分量の診療ガイドラインを読まなくとも、診療ガイドラインの内容を理解する助けになることが期待される。それにより、自身の受けている診療内容が診療ガイドラインに即しているか否かを確認することが可能となり、主治医とのコミュニケーション向上などを通じて、受療行為の改善に利することも期待される。

では診療ガイドラインをどの様にして電子化し、電子的な知識表現形式を構築すれば良いのか。診療ガイドラインの内容は端的に言って、「どのような状況では何をするべきか」という条件判断と手順を組み合わせたものであると言える。条件判断には、症状・徴候や検査結果、既往歴など診療上のデータが必要となる。こうしたデータには疾患名、所見、検査・処置の名称や処方薬剤名などが含まれているが、特に同義語や多義語の存在は混乱を招く。従ってデータとして使われる用語は、診療ガイドラインに記載されたままの名称ではなく、病名マスタや検査・薬剤名マスタなどの用語集との対応を取るべきである。これが診療ガイドラインの基底層の記述に相当する。また、条件判断においては、これらデータの組み合わせから得られる条件式が真になったときに、次の状態あるいは処置に進む、という構造が基本となり、これが手順層の記述に相当する。

A.2.　電子的記述の改善事項

これまで、本研究ではテキストで記載された診療ガイドラインを、GLIF(GuideLine Interchange Format)の形式に準拠した状態・医療行為・判断などのノードから構成されるフローチャート形式に変換し、同時に変換に際して曖昧な、もしくは明確に記載されていない項目や暗黙的な仮定などを明確に表現してきた。次に状態・医療行為・判断などを記述論理の形式で表現し、条件判断が真になる場合に次手順に進む形式を記述した。条件判断には診療上のデータが必要となるため、診療ガイドラインに記載された判断を行うために必要なデータが、実際の診療情報システムではどの程度カバーされているかを調査した。その結果、診療ガイドラインの判断に必要な各データ項目は、より基本的なデータ項目の組み合わせで表現できる場合があること、すなわち医療知識や常識を加えて基本的なデータ項目を組み合わせることによって導出されうることが示された。そこで条件判断に必要な情報をどのように導出するか、というルールも記述することによって、ある程度は診療ガイドラインの条件判断を評価することが可能となった。

しかし判断に必要なデータが得られない状況は、ある程度避けられないものである。そして判断支援に必要なデータが存在しない時には、一切の判断は不可能となってしまう。そこでデータが存在しない場合に推論を利用することによる限定的な判断支援を行うことに関する検討を行った。データが存在しない場合に、判断の結果が真になる場合と偽になる場合の両方を試行するアプローチにより、可能な場合を提示するという手法であった。その結果、真偽値の両方を不定と捉えてテストするという手法自体は有効である可能性があり、より適切な状況では良い結果が得られると考えることができた。そこで、データが存在しない場合、つまりデータの値が不明である場合を考慮した条件判断について、より詳細に検討を行う必要があると考えられた。

B. 方法

B.1. 従来の電子的記述

MINDS(財団法人日本医療機能評価機構) で公開されている高血圧診療ガイドライン、テキストで記述された診療ガイドラインを材料として、これまでに構築した高血圧診療手順の電子的記述をもとに改善を行った。この電子的記述は以下の様にして構築されたものである。まずテキストで記述された診療手順をフローチャート形式に変換した。同時に変換に際して曖昧な、もしくは明確に記載されていない項目や暗黙的な仮定などを明確に表現した。フローチャートを構成するノードは「状態」と「分岐」であり、各ノード間は「次手順」を示す矢印により、時間的順序に従って結ばれたものであった。

ここで「状態」には「外来受診時である」や「ある疾患に罹患している」などの状態を表すノードだけではなく、「血液検査を行う」や「降圧剤の処方を行う」などの医療行為を表すノードも含まれていた。そのため、介入内容を明確にデータとして利用することが困難であった。

知識表現には簡易記法を導入し、RacerPro の instance(個物) や related(関連) を単純に↑や⇔に置換することにより、直感的に解りやすくすると同時に文書量を削減して可読性を上げることに貢献した。しかし、実質的に手順を追う際には、related の記号表現である⇔に has-next が使われている行を注意深く追っていく必要があり、この点で手順を記述する際に注意を要した。また分岐ノードには、評価を行うために必要なデータが規定されていたが、それだけではなく判断ルールをも包含していた。これ自体は知識記述として判断のロジックを明確にするという点で優れていたと言える。しかし判断

ルールの中に LISP の演算子を使わざるを得ないこともあり、プログラム言語から独立した記述にはなっていなかった。さらに判断ルールには、判断ルールの評価によって分岐条件が真となったときには次のどのノードに分岐するか、まで記述されていたため、手順の部分と条件判断のロジックが混在してしまい、可読性を著しく損なっていた。

B.2. 電子的記述の改善

以上の問題点を考慮して、まず医療行為、特に処方・検査・手術・指導などの医療介入行為の内容を「状態」として終わらせるのではなく、介入内容をも個々にデータとして残すことができるように、これらを分離した。

次に、知識記述を読む際に、まず診療手順を追うことが容易であることを重視して、related の動詞を最初に記述することとした。また↑や⇔などの記号の導入は、文書量を少なくする反面、理解の妨げになるという面もあった。そこで、単純に instance や related などのような知識記述における原始的な文法を強要し、それを単純な記号に置き換えるのではなく、手順記述に必要な動詞を選別して定義し、これらを使って積極的に次の手順やノードが要求するデータ、そしてそれらのデータの属性を記述する方法を検討した。

また判断ルールについては、詳細な判断ルール記述が可読性を損なうことから、これを分離することとし、判断は単純に真偽判定で行い、個々の真偽判定を行うための関数名を明記することとした。それらの真偽関数のロジック詳細はコメントに記述し、それを実装するプログラミング言語に応じてロジックを実装すればよいこととした。真偽関数のロジックを文章で表現したのでは、元のテキストへの逆戻りとも見える。しかし判断にはどのようなデータを使用するのかを明確にした上で、それらのデータの組み合わせで判断ルールを記述しているため、単なるテキストとは異なり、むしろプログラムコーディングが容易になることを考慮した。

本研究では LISP を利用して処理系を構築しているため、真偽値を与える関数ロジックの記述にも LISP を使用している。実際、LISP に慣れてしまえば、新たな知識記述方法を構築するよりも、直接 LISP の構文を利用する方が開発においても、アルゴリズムを理解する際にも、理解が速いであろう。これは一般にプログラミング言語について言えることである。そしてひとつのプログラミング言語で記述されたロジックは、比較的容易に他のプログラミング言語に変換することができる。従って LISP で記述することのメリットは大きいと考えられた。

また昨年度までは知識表現の形式および実行するためのエンジンとして RacerPro を利用してきたが、これを使用する意義は知識記述の整合性を確認することや、知識記述に使われている階層関係をより単純な構造に変換することにある。しかし診療ガイドラインにおいては、知識構造の階層関係すなわち知識構造を記述するための語彙に関する階層関係を整備する段階にまでは至っておらず、むしろ診療手順を記述するために各診療状態のノードの間を has-next などの動詞で関連づけるに留まっていた。さらに RacerPro 自体は高価なソフトウェアであり、研究成果を配布する際には障害となることが予想された。以上より、これまでの RacerPro による知識記述を、より一般的な RDF 形式、あるいは n-triple 形式である AllegroGraph 形式に書き換えることを検討した。RDF 形式とは簡単に言うと「AのBはCである」という記述法である。例えば「自動車の燃料はガソリンである

」「坊ちゃんの作者は夏目漱石である」などのように、A・B・Cはそれぞれ主語・述語・目的語と考えることができる。そしてRacerProの記述では述語がrelatedで関連付けられる動詞に相当する事になるので、変換は容易と考えられた。

B.3.　条件判断に関する検討

条件判断において、条件を規定する値が真偽値の場合、存在しないことがすなわち偽である、とされることが多いが、これは真の値を取る場合だけを重要視したためである。偽を重視するのであれば、データが存在しない場合を真に仮定することもある。いずれも論理的には省略あるいは暗黙的な帰結が仮定されているものと考えることができ、そのままでは論理的に明確であるとは言えない。少なくとも真も偽も重視するのであれば、データが存在しない「不明」の状態を明白に持つべきであると考えられた。各ノードの状態も、真であるか偽であるか、に加えて不定である場合がある。つまり判断分岐を行うために必要なデータが存在していなければ、真偽の判定ができないことになり、その状態は不定である、とするしかないからである。真ではないから全て偽、では本当に偽であったものとの区別ができないことになる。従って、データの値が不定である場合、その結果としてノードの状態が不定である場合を検討対象に含めた。

また「どのような状況では何をするべきか」という表現を簡単に「AならばB」と書くことにすると、この「ならば」には多くの曖昧さが含まれている。基本的な意味は「Aが真であるならBも真」であるが、その時に「Bが真ならばAも真」というように、逆が成立する場合と成立しない場合がある。逆が成立する場合は必要十分条件、成立しない場合は十分条件である。

例えば「収縮期血圧が160以上180未満であれば中等度高血圧(JSH2009ではⅡ度高血圧)」において、与えられた血圧の値と中等度高血圧であることとは必要十分である。これは定義なので当然であるが「A＝B」に相当する。しかし2次性高血圧を除外する際に「血圧に左右差があれば血管性高血圧を疑う」という判断が存在するが、これについてはどのように考えるべきだろうか。

通常の条件判断ではここまで厳密に記述することはない。しかし、ガイドラインの中で限られた条件を基に探索を行う場合、例えばあるノードが成立しうる状況には何がありうるのか、を探索する際には、こうした厳密な条件記述が必要となることが考えられた。また、あるデータの値が他のデータと関連する場合、例えば概念上の階層関係があるような場合には、入力されたデータに基づいて、他のデータの値を自動的に設定することにより、ユーザーの入力負荷を減ずることになる。このような用途に対しても、データや判断ロジックの記述を検討することは重要であると考えられた。

また同じ「ならば」であっても、禁忌医療行為には「絶対に」が付されるのに対し、推奨の場合には「～が勧められるが、それ以外の場合も許容する」という意味が込められている。この両者には必然と可能性の違いがあり、確実に区別されるべきものと考えられた。

ところで、この後者の「ならば」は医療においては頻繁に出現する。例えば「頭痛があれば風邪の可能性がある」では、頭痛を呈する疾患の中には風邪がある、しかし頭痛は風邪の症状のひとつに過ぎず、頭痛があるからと言って必ずしも風邪であるとは言えない。医師は頭痛を呈する様々な疾患を考慮し、それらの疾患の

いずれであれば現在の状態を最も良く説明できるだろうか、という判断を行っている。このように、最終的な判断はロジックによって行うことはできず、人間の判断に委ねられるべきであるものの、人間の判断を支援するのがロジックの役割であると考えられる。そこで、こうした観点を考慮して、診療ガイドラインの電子的知識記述における条件判断を検討することが重要と考えられた。
纏めると、条件判断に関して、不定データに関する扱い、必要十分条件と十分条件の区別、必然と可能性の区別、について検討を行った。

B.4. 診療スレッド

診療行為は、ある状況下で与えられた問題に対し、何らかの意図を持ってこれを解決しようとする行為である。そこには現在の状況である出発点を規定する条件と、解決時の状況である終着点を規定する条件が存在する。ここで意図実現を達成する方法には複数の手段がありうる。ひとつの意図に対しては様々なアプローチが存在し、その中で最も有望なものを選択して解決しようとするが、もし解決できなければ別のアプローチを試行しなおす。医療においては不確定な状況が多々存在するため、このような試行過程を考慮することは重要であると言えるし、治療効果と費用あるいは生活の質などのように、単一の価値観では単純に比較できない場合があるため、多くの異なった種類の問題解決方法を考慮しなくてはならない。これら診療の方向性を「診療ベクトル」と呼ぶが、その方向性に沿って目標が達成されたか、達成されなかったので破棄するか，という診療単位の連なりを診療スレッドと呼ぶ。例えばアキレス腱断裂が生じて、これを治療する場合を考えてみる。状況はアキレス腱が断裂していること、意図はアキレス腱の癒合および元通りに運動できるようにすること。それに対して治療期間、費用、入院か外来か、再発の可能性を考慮して保存的か外科的かなど、様々な介入方法が存在し、個々の患者に対して、これらの選択のいずれが最適であるか、を決定しなくてはならない。診療手順の知識を記述する際には、こうした要件も考慮しておきたい。
診療ガイドラインは通常ひとつの疾患に対する可能な手順を提示するものであるが、現実の臨床においては上記の例のように、多くの疾患や状況を考慮しなくてはならない。すなわち診療スレッドに関わるような、現実の複雑な診療を支援するためには、単一の診療ガイドラインの知識だけでは不足であることになる。しかしながら、複数の診療ガイドラインの知識やその他の一般的な知識記述を総合することにより、こうした現実の状況に対する人間の判断を支援しうる可能性がある。こうした応用を考えると、診療ガイドラインの知識記述における「状態」を特定する条件が記述できれば、様々な診療ガイドラインを跨った検索を行うことによって診療スレッドにおける判断支援を行える可能性があるため、検討対象とした。

B.5. 診療手順の区切りとしての診療ブロック

診療スレッドにおいては、問題点である現在の状況と解決点である目的地の状況の特定が重要であった。診療ガイドラインの手順も、診療に立脚するものであるため、こうした現状や目的に従っていくつかの部分に分けることができる。通常は 1 回の外来診察の中に収まることが多く、状況や意図を同じくする一連の手順の集合として切り分けることができる。例えば高血圧

の原因には何があるのか、どのように血圧をコントロールしたら良いか、血圧コントロールは維持できているだろうか、などであり、それぞれ高血圧の診断、治療、経過観察に相当し、それらはより基本的な手順要素から構成される。こうしたひとかたまりの手順集合を診療ブロックと呼ぶ。これは診療スレッドの問題 / 前提条件と目的 / 終了条件を抜き出して、診療ガイドラインの手順に当て嵌めたものである。この診療ブロックという分類を記述することにより、診療ガイドラインに記載された診療手順全体のどの段階にいるのかが明確になり、各ブロックの初期条件と終了条件を明確に記述することが可能となると考えられた。

(倫理面への配慮)
本研究では特定の患者に関連する情報や個人識別情報は扱っていないため、倫理的側面は存在しない。

C. 結果

高血圧診療ガイドラインの手順を電子的記述に変換したものの主要部分を資料 2 に示す。ここには手順を記述した部分だけではなく、条件判断の各関数の LISP による定義も含まれている。

C.1. 電子的記述の改善

診療ガイドラインの手順を記述する知識表現の形式を修正し、以下の様な記述形式とした。
まずノードAの次手順がノードBであることは、
(has-next A B)
によって記述した。これにより「次手順」を示す動詞を明確にした。
ノードBに以下の判断分岐に必要なデータがある場合には、
(has-data B data-1 data-2 data-3)
のように記述することにより、ある手順ノードで用意すべきデータとその属性を明確に記述した。
データの属性の指定には has-type を使い、integer、float、string、list によりデータ固有の属性を指定することができることとした。それぞれ整数、実数、文字列、リストを意味し、例えば data-1 が整数であれば
(has-type data-1 integer)
のように記述する。ここでリストとは幾つかの string を空白で区切り、全体を括弧にくくったものであり、その使い方は LISP に準拠してこの list を利用する関数によって解釈されるとした。
例えば初診時診察で診察所見というノードでデータを収集するには
(has-data 診察所見 年齢)
などを定義すれば良く、さらに

(has-type 年齢 integer)
という記述によって年齢が整数であることを宣言するが、同時に整数データが存在するか否かも内部的には記録できるようにした。つまりデータを入力する場面で、年齢が入力されればデータが存在し、入力されなければデータが存在しないために不定である、という扱いをすることにした。これは実際には後に述べる「年齢」を条件判断に利用する関数の中に記述することによって実現した。
has-type には、この他に selection と radio がある。selection は以下のリストの中からひとつ以上のものが選択されうること、そして radio は以下のリストの中からひとつだけが選択されうることを意味することとした。例えば
(has-type 心疾患 selection ない 左室肥大 狭心症 心筋梗塞 心不全)
では、心疾患には selection 以下のリストの中から複数の値を取りうることを示しており、狭心症が真になれば、心疾患も真になる場合に使用された。ここで「ない」はいずれの疾患も存在しないことを意味しており、これが指定されれば心疾患というデータが確実に存在し、その値が偽であることになる。もしこれらのいずれもが指定されなければ、データは取得されていないため不定となり、心疾患というデータ自体が真とも偽とも評価できないことになると解釈された。
条件分岐は、複数の has-next を記述した上で、各分岐先のノードが選択されるための関数名を has-predicate によって指定することにした。
例えば
(has-next A B)
(has-next A C)
(has-next A D)
(has-predicate B function-b-p)
(has-predicate C function-c-p)
(has-predicate D function-d-p)
の様に記述することによって、ノードAからB・C・Dに分岐すること、各分岐のいずれが選択されるかは、それぞれのノードに付した関数である function-b-p・function-c-p・function-d-p の中で真になったものが選択されるものとした。この時、各関数で評価されるデータが不定である場合には、関数は真を返すため、どの分岐も選択されることになるものとした。

C.2. 条件判断に関する検討

「AならばBである」という条件判断には、方法に記述した様に、様々な曖昧性が含まれている。これを安易にプログラム言語で記述するのは曖昧性を区別することによる複雑さを増大させるため、現状では得策ではないと考えられた。むしろ判断ルールの内容はコメントとして詳細に文章で記述されているので、上記のような分岐関数を固有のプログラミング言語で（本システムの場合は LISP である）実装する際に注意することで曖昧性を解消することができた。すなわち診療手順の電子的記述を行う際に、必要十分であることをコメントに明記しておくことが重要であった。これは例えば「収縮期血圧が 160 以上 180 未満であれば中等度高血圧 (JSH2009 ではⅡ度高血圧）である」場合に相当した。
その一方、「頭痛があれば風邪の可能性がある」や「血圧に左右差があれば血管性高血圧を疑う」は曖昧である。頭痛を呈する疾患の中には風邪がある、しかし頭痛は風邪の症状のひとつに過ぎず、頭痛があるからと言って必ずしも風邪であるとは限らず、くも膜下出血があるのかもしれない。医師は頭痛を呈する様々な疾患を考慮し、それらの疾患のいずれであれば現在の状態を最も良く説明できるだろうか、という判

断を行っている。ところで、風邪であっても必ずしも頭痛を生ずるとは限らず、咳しか生じない合もあるかもしれないことに注意すると、ここには論理学的な意味での因果関係は存在せず、単に共起関係があるだけであると言える。しかし日常生活でもガイドラインの記述でも「AならばB」という形式で捉えられることが多い。これは「可能性がある」程度に留めておくべき要件であり、最終的な判断は人間に確認を求めるべき性質のものであると考えられた。

しかしこのような「可能性がある」場合であっても、判断ルールとしては手順を分岐するために利用しなくてはならない。そこで、分岐する前にその評価結果を人間に提示して最終的判断を促すようにした。例えば2次性高血圧の原因疾患のひとつである血管性高血圧の場合、これが疑われる症状のひとつに「血圧の左右差」がある。そこで「血圧の左右差」をデータとして入力させてこれが真になった場合、つまり確実に血圧の左右差が存在する、とされた場合であっても、その次にあえて「血管性高血圧」という疾患を、それがチェックされた状態で提示することにより、人間に確認を求める方式を採用した。もし血管性高血圧が本当に疑われるのであれば、これを真のままにして残せば良いし、血圧の左右差はあるが血管性高血圧よりも本態性高血圧が疑われるのであれば、このチェックを外せば良いからである。

禁忌については、手順としての記述が難しいことは前年度に考察した。そこで「AとB」が禁忌である際に、それが同時に選択された状況が生じたならばその手順を許容しない、という解釈を行うことにより、手順として、特に降圧薬処方の経路に実装した。

C.3. 診療ブロック

臨床思考過程モデルを基礎に据えた診療スレッド、すなわち治療方針という意図を実現するために種々の仮説を検討すること、そして仮説が棄却された場合に意図は同じであっても異なる実現過程を新たに選択すること、という診療スレッドに関する知識記述への実装については検討を重ねたが、これを知識記述というレベルで扱うことは非常に複雑であると考えられたため、診療ブロックとは分離させた。将来的に診療スレッドの知識記述を扱う際には、各診療ブロックの前提条件と目標を、何らかの方法で定式化することによって、複数の診療ガイドラインや診療知識と機械処理を通じて連携させることが可能と考えられるが、現状の知識記述の枠組みを逸脱するものと考えられた。

次に診療の基本手順を組み合わせた診療ブロックを明確にした。高血圧診療を例に取ると、診断・治療・経過観察である。診断も本態性高血圧の診断と 2 次性高血圧の診断などに分離可能であるが、状況や意図を同じくする最も大きなかたまりをブロックと規定したため、上記のように分けられた。

これら診療ブロックに関して、has-block、has-requirement、has-goal、has-node という動詞を導入した。(has-block 高血圧診療ガイドライン 高血圧治療ブロック) によって高血圧診療ガイドラインには高血圧治療ブロックが存在することを記述し、(has-requirement 高血圧治療ブロック 目標血圧の設定) によって治療ブロックに入る前提条件として目標血圧が設定されていることを記述、(has-goal 高血圧治療ブロック 降圧薬処方) によって高血圧治療ブロックの目的を記述、そして (has-node 高血圧診断ブロック 本態性高血圧) のように診療ブロックに属する

手順ノードを個々に記述することができた。

D．考察

D.1.　手順層の記述

本研究では、電子的記述を大幅に見直し、has-next、has-data、has-type および分岐先を指定するための関数名を導入した。こうした動詞で記述された知識は、マクロによって n-triple の記述に変換される。以前は RacerPro などの基本的な言語の仕様に従って記述しており、これは本研究ならば直接 n-triple で記述することに相当する。このため、手順の記述自体は大幅に容易になった。しかしテキストで記述された診療ガイドラインを直接、この形式に変換するのは不可能、少なくともフローチャートを作成した後に、それを対照しながら、個々に手順の記述を実施したが、それでも多大な手間と時間を要した。特に、いったん (has-next A B) の形式に記述してしまったものを修正する場合が面倒であり、常に基となるフローチャートを見直して修正しながらでなければ作業が進められなかった。電子的記述の改善により、可読性が増しているのは確かであったが、電子的記述は多数の分岐が 1 行に展開されてしまうと同時に、個々のノードの手順を記述するだけではなく、ノードに与えるデータの記述や分岐させるための関数名の記述を行う際には、手順のどこにそれぞれのデータや関数が対応するのか、が見えにくくなるためであった。

ところで、UML で記述した図をもとに、自動的にクラスやメソッドを生成するツールが存在する。これと同じように、診療ガイドラインの手順をフローチャートとして VISIO などで描画したものを、自動的にここで定義した has-next、has-data や分岐先で真となる関数名による記述に変換するツールがあれば、大幅に開発効率が

改善すると考えられる。その実装は十分に可能と考えられ、今後の課題としたい。

D.2. 基底層の整備

本研究は手順層の記述を検討することが主目的であったため、基底層に関してはあまり言及してこなかった。しかし手順層を記述する際に、基底層の整備は極めて重要である。本研究で電子的に記述された知識を元の診療ガイドラインのテキストと比較すると、基底層は個々の単語に相当し、手順層は単語間や文章間のつなぎの部分である論理的な繋がりに相当する。

個々のノードの名称である診療状態や介入段階の名称はともかく、特に条件判断を行う際の材料として評価されるデータにおいて、基底層の整備は重要である。これらのデータあるいは医療用語には、データの名称とその値があり、それらは正確に区別されるべきである。医療用語（の名称）は、標準病名集や MEDIS で整備されているマスタ類が対応する。これらには同義語もあるため、同義語が存在する場合には代表語を使用するか、あるいはマスタ上でのコードを（もし存在するならば）使用するべきである。またデータとして利用される際に取り得る値には何が許容されうるのか、も規定しておく必要がある。一般に用語集や用語マスタでは、その用語が使用される環境が暗黙的に仮定されている。例えば病名集は病名欄に使われるであろうし、薬剤名は処方欄に書かれることが大前提である。従って用語だけを定義することが重要であり、その値に意識が及ぶことはない。しかし電子的にデータを利用する際には、データの名称と値の両方が重要である。例えば「収縮期血圧」であれば、その値は整数であり単位は mmHg であるし、「既往歴」であればその値は疾患名のリストであるべきなのである。

このように、使用するデータの仕様が定まれば、診療ガイドラインを電子化する際には実装が極めて容易になる。実は診療情報システムにおいても、それは全く同様であると言える。

しかし現在のところ、医療分野全体で使用可能な、この様な用語集は存在しない。またこのような膨大なリソースが構築されるのを待つのも得策ではない可能性が高い。何故ならそのような用語集は網羅性を目指すために個別の応用面、あるいは目的が見えにくくなるためである。むしろ、各応用分野で独自の用語集を構築、利用しているのが現状であるため、それらの間で矛盾が生じないように調整することから始めることが得策と考えられる。従って、診療ガイドラインにおいても、高血圧診療ガイドラインの用語集（とその値の規程）、糖尿病ガイドラインの用語集などを電子化する際に、個別に構築したものを比較し調整することによって、徐々に語彙を拡充していくのが良いのではないだろうか。

複数の用語集を調整する際には、個々の用語がどの分野・状況で使用されているのかを正確に把握するべきであろう。異なる分野間で矛盾が生じたことを検出することが容易になり、誰に調整を依頼するべきかが明らかになるからであるし、状況を把握することにより、その用語が使われているコンテクストの違いを考慮することができるからである。逆に、調整の結果として、ある用語が変更された際には、その用語を使用している分野側でも迅速に用語を更新することが可能となる。

この様に基底層の整備ができていないことは日本の医療全体の問題になりつつある。これまで各ベンダーは診療情報システムにおいて、施設毎に必要とされた個別カスタマイズに対して多大な費用を費やしながらも対応することはでき

た。しかし診療報酬請求でICD-10に準拠した病名集が共通に扱えるようになる必要性があったのと同じように、診療ガイドラインやそれを元にした判断支援など、今後様々な医療リソースを日本全国で利用することができるようにするためには、やはり基本となる医療用語の整備が必要なのではないだろうか。

D.3. 条件判断に関する検討

論理学的には、Aが偽である時にBの真偽に関わらず常にこの命題は成立する。直感的には「「AならばBである」の否定」を「「Aであるのに Bではない」の否定」と考えれば理解しやすい。後者は「Aかつ「Bでない」」の否定なので、「「Aでない」またはB」となり、ここでAが偽ならば「Aでない」は真になるため、Bの真偽に関係なく全体は真になるからである。これをガイドラインの条件判断に適用すると大変困ったことになる。「Aが成り立たない時はBでもBでなくとも良い」を先の中等度高血圧にあてはめると「収縮期血圧が120の時には、中等度高血圧であるとも中等度高血圧でないとも言える」となる。これは確かに間違ってはいない。しかし無意味な陳述でもあり、混乱を招く。

これは「AならばB」を論理包含(あるいは含意)として厳密に考えた場合であり、A⇒Bと表現される。それに対して通常の「Aが真ならBも真」はA→Bと表現される。この両者にある隔たりあるいは乖離は「実質含意のパラドクス」(Anderson and Belnap 1975)として知られており、論理学の世界では適切さの論理、あるいは相関論理(Relevance logics)で扱われているものである。しかしこれを診療ガイドラインのような実務に応用するのは時期尚早と言える。まずは、本研究の結果で述べたように、必要十分であるのかないのかを意識した上で、具体的な事例に則して条件判断のルールを記述することが重要と考えられる。

また、条件判断では、条件を満たした場合には必ず次の状態に至る「必然」の場合と、条件を満たしていれば次の状態に至りうる「可能性」の場合とがある。これは時間的推移に対しても当てはまり、検査結果が陽性であれば必然であるが未定の場合には可能性にとどまる、という立場がある。こうした状況はいずれも様相論理によって扱うことができる。

様相論理では必然性演算子である□と可能性演算子である◇を使う。

例えば　　$\Box p \Leftrightarrow \sim\Diamond\sim p$

という記法により「必然的な真」は「偽である可能性がない」、あるいは「必然的な偽」は「真である可能性がない」が成立する。

また　　$\Diamond p \Leftrightarrow \sim\Box\sim p$

という記法により、「真である可能性がある」は「必然的に偽であるわけではない」、あるいは「偽である可能性がある」は「絶対に真であるとは限らない」と同等である。

様相論理には様々な流派が存在する。また未来演算子や過去演算子を加えた時制論理も存在する。これらを利用して推論を行うことも研究課題としては興味深いところである。しかし本研究では診療ガイドラインにおける手順の提示や検索を行うことが目的であり、判断の主体は人間であるとの方針から、何もかもコンピュータで推論をさせることが主目的とはならない。むしろ様相論理という考え方が存在し、その背景に必然と可能性という区別が論理においても認識においても存在するということを意識した上で、通常の「AならばB」という条件判断を解釈することが重要であると考える。

ところで、分岐において分岐条件が真となった

結果、has-next で指定されるノードは真となる。では他のノードは偽となるのだろうか。これも問題を含むことは明らかである。分岐が横並びに幾つかあった場合、個々の分岐条件が従属的であるとは限らないからであり、幾つかの分岐が同時に真となるかもしれないことを考慮しておく必要がある。理想的には、ひとつの判断から分岐するノードは互いに素であるべきで、ひとつが真ならば必ず他は偽となるべきなのである。すなわち、条件判断においては必ず従属関係にある分岐のみでひとつの判断を構成するべきである。そして従属ではない独立な条件が存在する場合には、判断を 2 つ以上に分離すべきである、という事になる。

いずれにしろ、テキストで記述された診療ガイドラインから「AならばB」のような関係を抽出してフローチャートに変換する際には、こうした論理包含や含意と日常の「ならば」との違いや、可能性と必然、それも時間的な推移も混在した複雑な関係を意識した上で、適切に解釈することが重要となる。メディアリテラシーという言葉があるが、それを借りるならばロジカルリテラシーとでも呼べるであろうか。極めて単純な例をあげると、JSH2009 に「血糖値異常（空腹時血糖 110-125mg/dL、かつ / または 糖尿病に至らない耐糖能異常）」という記述が存在する。この「A かつ / または B」とは「AとBのいずれかがある場合も、両者がある場合も」という意味であろうが、これは論理的には「AまたはB」で十分である。しかしこのように記載すると逆に「AかつB」が生じた場合が示されていない、と感じる読者が存在するのかもしれない。

従って、診療ガイドラインのテキストには、文章での記載とともにフローチャートでの記載を求めるべきなのかもしれない。そのフローチャートには、必要十分であるのか、可能性があるレベルなのか、共起なのか、あるいは機械的に帰結させて良い分岐条件であるのか、人間による最終判断を要するのか、という項目あるいは記号が導入されているべきであるが、そのようなフローチャートによる記載も同時にされているのであれば、読者に与える（論理的な）誤解も最小限に抑えられるであろうし、本研究のような電子的記述への変換も極めて容易となることが予想される。

D.4. 診療スレッドと診療ブロック

診療スレッドに関する知識記述への実装は、結果に述べたように本研究の範疇では困難と考えられた。本報告書の「診療手順の知識記述における診療スレッドの実装に関する研究」では、高血圧治療に対する異なる方針が存在する場合を想定して、循環血漿量を減少させる方針と末梢血管抵抗を減少させる方針の 2 つの目標を想定した。そして各降圧薬の特性として、これらの方針にいずれかに属するか否か、を追加することによって、両者の治療方法の経路を切り替えることができた。これも診療スレッドの実装のひとつではあったが、一般化は困難であった。また前提条件や終了条件を使って類似したものを検索することも難しい。さらに、様々な診療ガイドラインや価値観に跨って統一的に記述することも難しい。人間が診療において様々な可能性に思いを巡らせる、という状況を、手順記述によって実現するためには、このように個々の診療ブロックにおいて、診療場面で出現しうるありとあらゆる様々な可能性を実際に記述しておかなくてはならないことを意味するからである。しかし診療ブロックに関する実装は、定型的には一応満足できるものであった。しかし記述することは可能であっても、それを実際に応用す

る際には、特に経過観察ブロックに関して問題が生じた。これについては応用だけではなく、知識記述の手法に関しても考察を進めているため、「診療ガイドライン手順知識の提示　－対話的操作可能な手順グラフ－」を参照されたい。

E．結論

診療ガイドラインの手順内容を電子的に記述した場合に、材料である用語を基底層、手順の連なりや「AならばB」のような論理的な関係を手順層、そして得られた電子的記述を用いて診療などの場に利用する部分を応用層と呼ぶと、本研究では従来の記述法における反省点を踏まえて、特に手順層に関する検討を行った。記述法を改良することにより、手順に含まれる状態や行為、条件分岐で評価されるデータ、目的を同じくする一連の手順の集合である診療ブロックに対する記述が容易にできるようになり、可読性を高めることができた。またAならばBという条件判断について検討を行い、論理的な含意と実生活での「ならば」との乖離をはじめとして、可能性と必然の区別、因果関係と共起の区別などを意識する必要性を意識した上で判断ルールを記述することが重要であり、最終的に人間による判断を仰ぐための仕組みを導入する必要性について論じた。

F. 研究発表

1. 論文発表

2. 学会発表

[1] 小野木雄三　廣瀬康行．診療ガイドラインと診療スレッドの知識表現．医療情報学．28S: 1092-1097. 2008

[2] 廣瀬康行 小野木雄三．医療介入に関わる知識表現への診療スレッドの応用．医療情報学．28S: 1087-1091. 2008

G. 知的財産権の出願登録状況

1. 特許取得

なし

2. 実用新案登録

なし

3. その他

なし

診療ガイドラインによる診療内容確認に関する研究

診療ガイドライン手順知識の提示 ー対話的操作可能な手順グラフー

研究要旨　：　診療ガイドラインの手順を記述した電子的知識表現をもとにして、手順グラフという形式で表示することにより、ガイドラインに記された診療手順の内容を視覚的に解りやすく表現した。同時に対話的なデータ入力によってグラフの形態をダイナミックに変化させることを可能とした。これは診療における判断支援に利用することができるだけではなく、仮想的に様々なデータを入力することによってどのような帰結が得られるのかを試すことにより、診療ガイドラインの内容をより深く理解することにも利用することができた。最後にこのシステムの開発を通じて得られた知見について考察を行った。

A. 目的

診療ガイドラインは通常、テキストとして与えられている。ある疾患に関する病態の解説、疫学的な解説、原因やリスクに関する解説などは、その疾患の全般的な理解を深めるために重要な項目である。しかし診療ガイドラインとして最も重要な項目は、その疾患が疑われた場合にどのように診断するか、診断が成された時にどのようにして治療に当たるのか、そして経過観察が必要な場合にどのようにしてそれに処するのか、といった事項であり、それらが文献的な根拠に基づいて記述されているところに意義がある。しかしそうした事項がテキストで記述されているのみでは、理解が容易ではない。実際、市販されている診療ガイドラインを一読すればわかるように、多くの診療ガイドラインでは理解を容易にするために、手順を示すダイアグラムや重要な事項を簡潔にまとめた表形式などが添えられている。

これまでの研究では、テキストで記述された診療ガイドラインから手順を抽出し、電子的な知識形式に記述し直すことにより、判断支援システムなどへの応用を行うことを目標としてきた。判断支援システムとは、与えられた状況下で人間が行うべき判断を支援するものである。電子カルテやオーダリングシステムにおいて、何らかの指示が成された時に、システムが自動的にデータや知識をチェックすることにより、誤った指示が成されるのを未然に防いだり、人間の求めに応じて様々な助言を行ったりすることが可能となる。

このような判断支援を行うためには、判断の根拠となるデータが必須である。しかし臨床の場では必ずしも判断に必要なデータがシステム上に存在しないことがあり、判断支援を行うことが難しいという問題がある。

ところで診療ガイドラインの知識は、判断支援システムとして利用するだけではなく、診療ガイドラインの手順全体を提示することによって、その内容の理解を深めるためにも利用することができる。そこで本研究では、診療ガイドラインの内容をより容易に理解することを目指し、電

子的に記述された診療ガイドライン知識を元にして、手順をわかりやすい形式で提示することを第１の目標とした。

ここで、ガイドラインの手順を冊子版のフローチャートや表形式で提示したのでは、わざわざ電子的に記述する意義はない。静的で変化しないダイアグラムを提示するのではなく、むしろ具体的なデータが与えられたときに、どのような手順を取るべきかが明瞭に示されるようなインターフェイスによって手順を提示することが望まれた。

また判断に必要なデータが欠落していると、条件判断を正確に実行できないため、これをそのまま全体手順の表示に応用しようとしても、データが欠落した判断分岐以降のパスは表示できない事になる。そこで前年度の研究で行ったように、欠落データが存在した場合に、そのデータに依存する条件判断の真偽値を両方テストすることにより、存在しうるパスの探索を行う手法が利用できると考えられた。

以上により、診療ガイドラインの知識を利用して医師や患者を支援するために、欠落情報があることを前提としながらも、診療手順を解りやすい形式で提示することを目的とした。テキストで記述された診療ガイドラインを電子的知識表現として記述する際に使われる材料としての医療用語を整備する領域を基底層、知識を記述する領域を知識記述層とすると、これは電子的知識を利用する応用層の検討であると言える。ここでは特に臨床的あるいは実用的な見地を重視しながら、電子的知識の応用層について検討することを目的とした。

B. 方法

B.1. 診療ガイドライン手順グラフ

診療ガイドラインの手順を上位から順に辿っていく方法として、フローチャートの形式で視覚化することは理解の助けになる。特に状態、実施項目、判断が明確に表示され、特に判断において必要なデータの組み合わせやそれらの値に応じて、どのような分岐に進むのかを追っていくことが容易であることが、理解しやすさのポイントと考えられる。そこで、視覚的にもフローチャートと類似したグラフ形式、あるいはツリー表示を採用し、最上位の「高血圧初診」という状態から順に何をしてどのように分岐していくのか、を辿ることのできるインターフェイスが有用と考えられた。

フローチャートを自動的に描画する手法として、Sugiyama モデルと Kamada-Kawai モデルを検討した。Sugiyama モデルは、グラフの表示エリアに磁場が存在し、各辺（エッジ）に置かれた磁性に応じて各ノードを配置する。また Kamada-Kawai モデルは Eades によるばねモデルを基礎にしている。ばねモデルは各エッジが自然長を持ったばねであると仮定することにより、エッジで接続されたノード間にはばねによる引力が働くと同時に逆 2 乗則による斥力が働くものとして各ノードを配置する。Kamada-Kawai モデルは、ノード間に働く逆 2 乗則の斥力を取り除き、その代わりに全てのノードがばねで結ばれているものとする。そしてそれらの各ばねは、ノード間のグラフ理論的な距離である最短パスの長さによって決まる自然長を持つものとする。ここで、Kamada-Kawai モデルでは上から下に階層的に配置されるように、同じ層のノード群に関してこのアルゴリズムを適用した。

次に条件判断の分岐に必要なデータは、分岐ノードでユーザーの入力によって得られるものとした。そして個々のデータ項目に対してユーザーから入力された値に応じて計算が行われ、どの分岐に落ちるのかが表示されることとした。このようにデータ入力に対応してグラフが対話的に即座に更新表示されることは、ユーザーインターフェイスとして非常に重要な項目であると考えられる。視覚的にフローチャートのような形式を使うこと、かつ対話的にデータ入力を行い、その結果がすぐにグラフに反映することを要件とした。

また分岐において選択されていない枝は表示されないか、暗くなって選択されていないことが明確にわかるようにするなど、選択の有無を表示する方法に関して検討を行った。

なお、ここではデータはユーザーからの入力を前提としている。もちろん電子カルテやオーダリングシステムからデータを取得することが可能ならば入力する手間を省くことができる。しかし得られたデータを元にどのような応用層の提示を行うのかを検討する本研究においては、データの入手方法や連携に関する項目は本質的ではないと考えられるため、これらに関する検討や開発は対象外とした。

ここで、入力データと別の入力データとの間には依存関係が存在することがある。例えば 2 次性高血圧の原因のひとつに血管性高血圧があり、その症状には左右の血圧差がある。もし血圧の左右差が存在することがわかれば、血管性高血圧が存在している可能性がある。また血管性高血圧が存在すれば 2 次性高血圧である可能性が高くなる。このように、データが相互に依存している場合、システムはあるデータに依存する他のデータの値を自動的に設定することができるようにすることが有用と考えられた。例えば「血圧の左右差がある」という入力が成されたならば、自動的に「血管性高血圧がある」というデータ項目をセットする。血圧の左右差を問うデータ入力と、血管性高血圧を問うデータ入力を 2 重に行わせる手間を省くためである。しかし血管の左右差があったとしても実際に血管性高血圧があるとは限らない。このように「～の可能性がある」という依存関係がある場合には、自動設定を行った上で、その結果を人間に提示し確認させることが重要であると考えられたため、検討対象とした。

また例えば、痛風や低K血症がある時にサイアザイド系利尿薬は禁忌である。従って、データ入力において任意の時点で痛風もしくは低K血症であることが設定されたなら、システムはサイアザイド系の利尿薬は処方することができないようにするべきである。これは可能性があるというレベルではなく禁止レベルなので、必ずしも人間に確認を求める必要はなく、むしろ積極的に警告を発するべきである。いずれにしろ、こうした依存関係こそが知識であり、それを支援することがシステムの役割であり、これを重視してインターフェイス設計を行った。

材料として、高血圧診療ガイドラインをもとに作成した知識表現をもとに、初診時から順に診断、治療、経過観察と手順を辿るインターフェイスを構築した。開発環境は Allegro Common LISP、知識表現は Allegro Graph の n-triple を利用して構築し、システム開発を行った。

前年度まで、電子的な知識記述を行う platoform として RacerPro を使用してきたが、今年度はこれに代わり Allegro Graph を使用することとした。その理由は手順を追うことと条件判断を記述するだけならば、膨大な知識記述を行った場合の整合性チェックや高度な推論処理といった機能は必要ないと考えられたからで

ある。例えば次手順を示すには、次ノードを指し示す has-next という関係を使い「Aノードの次はBノードである」を (has-next A B) という形式で RDF あるいは n-triple 形式で記述することができる。また条件判断についても、真偽で分岐するならば真偽値を返すひとつの関数とその否定、多分岐ならば真の時に分岐先に進むような関数群を記述すれば良い。この状況自体は RacerPro でも同様であったが、条件判断評価を実行する関数の実体を知識として記述していたため、非常に煩雑で理解が難しい記述にならざるを得なかった。その点、Allegro graph を使用するならば、最初からプログラミング言語として LISP を使用することが前提となり、関数記述もそのまま LISP で記述することができる。これにより、関数記述そのものも非常に簡潔になることが期待された。

データが数値の場合、値が入力されれば何らかの閾値で判定を行ったり、他の値と組み合わせて数式を計算したりすることによって条件判断が評価される。従ってデータが入力されていなければ評価が行えず、データが存在しないことは明らかである。しかし「喫煙の有無」や「腎疾患の有無」などのデータはデータが存在しないことが明確にはわからないことがある。真偽の 2 通りしかないので、データが存在しない場合は偽の値を取ると解釈、あるいは暗黙的に了解されてシステムが設計されることがあるからである。しかしこれではデータが存在しないのか、真もしくは偽の値を持っているのかを区別することができない。そこで本システムではデータの有無と真偽とを区別することにした。ただし、全てのデータに関してデータが存在するのか存在しないのか、存在するなら値を何であるのか、を入力するのは非常に煩雑になることが予想され、これを改善することも検討に加えた。

B.2.　診療ブロック

診療行為には意図が伴う。高血圧診療における意図は血圧のコントロールであるが、診療ガイドラインに従って手順を遂行していく過程で、その意図は変化していく。初診時には高血圧の原因を明らかにすることであり、本態性高血圧であると知れた後は血圧のコントロールを行うための生活指導や降圧薬処方によって高血圧の治療を行うことである。そして生活指導や降圧薬治療の効果を評価し、指導や処方の内容を微調整することによって、血圧のコントロールを維持する。高血圧診療ガイドラインの電子的知識記述では、この方針に従って全体の診療手順を「診断」「治療」「経過観察」の 3 個の診療ブロックに分割しており、本研究ではこれをそのまま利用した。

知識表現において、診療ブロックとは診療ガイドラインの手順ノードの繋がり全体から一部を切り出したものである。その入口である最初のノードに入るための条件が前提条件であった。また、出口に至るまでの条件が終了条件であり、これは同時にその診療ブロックの目的と考えることもできる。知識表現において、前提条件も終了条件もテキストで与えられている。つまり人間が読んで理解することはできるが、知識表現やプログラム言語では与えられておらず、あくまでも個々の診療ブロックの意義あるいは簡潔な説明を与えるものとして、ユーザーの理解を深めるために導入されたものであり、特に検索結果として得られたノードのリストを提示する際に、どの区画にあるノードであるのかを示すために有用と考えられたものであった。また、この診療ブロックの内部では、データや患者の状態が変化することはなく、単調な状態を維持するものとされた。これにより、診療ブロックの内部では矛盾を生じ