TR-661

仮説推論を用いた知識検証支援システム

田中　立二、石川　啓子、
山尾　雅利（東芝）

June, 1991

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03)3456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

人工知能学会

（投稿原稿：論文）

仮説推論を用いた知識検証支援システム

Knowledge Verification System with Assumption-based Reasoning

石川啓子*1
Keiko Ishikawa

田中立二*1
Tatsuji Tanaka

山尾雅利*2
Masatoshi Yamao

＊1　株式会社　東芝　重電技術研究所

Heavy Apparatus Engineering Laboratory, Toshiba Corporation

＊2　株式会社　東芝　府中工場　情報処理システム開発部

Computer Application Systems, Development Department,
Toshiba Corporation Fuchu Works

Keywords:　Knowledge Verification,
Knowledge base,
Assumption-based reasoning,
ATMS

Summary:

Knowledge acquisition and verification is a critical bottleneck of expert systems. In particular, it is difficult to confirm and maintain the consistency of a large-scale knowledge base. Knowledge verification is the process that makes problem solving knowledge complete and consistent.

KNOV (Knowledge Verification System) has been developed for diagnostic applications. KNOV is a meta-system which regards the diagnostic knowledge as an assumption and verifies it by assumption-based reasoning using ATMS. An architecture with the following features is proposed for knowledge verification:

1) assumption-based reasoning with dynamic testing
2) meta-knowledge definition for verification
3) knowledge consistency using ATMS

The validity and effectiveness of the ideas for knowledge verification were confirmed by applying KNOV to the diagnosis system of Electric Power Systems and Computer-Center Fault Recovery Systems used in the field. This paper shows a verification example using KNOV for the Electric Power System's diagnosis knowledge base. Last, problems to be solved for practical use of the system are pointed out.

1．まえがき

知識工学関連の研究開発が日本において盛んになり始めて、約10年が経過しようとしている。研究の対象は、簡単な推論システムから始まり、エキスパートシステム構築の環境全体へと発展しており、その産物として、現在、数多くのエキスパートシェルが開発されている。これらの基礎技術や方法論の発達につれて、エキスパートシステムを導入・実用化するケースが増えてきた。初期の頃のエキスパートシステムは、限られたルール数によりオフラインで動作するものであったが、次第に目的が多様化し、オンライン動作、あるいは外部ソフトウェアとの結合などが必要とされて来ており、こういったエキスパートシステムの知識ベースは、より大規模に、そして、より複雑化する傾向にある。ここで問題となるのは、知識ベースの管理である。エキスパートからの知識の抽出、抽出した知識の確認・整理・構造化は、知識が大量であるほど困難である。さらに、一旦、知識ベースを作り上げた後も、知識の追加、知識の修正を行う事は十分考えられるが、その場合、すでに存在する知識との整合性の維持は必要不可欠である[1,2]。知識獲得に関しては多くの支援システムが開発されているが[3-10]、知識検証を支援するシステムの研究開発は少ない[11,12,13]。

本論文では、以上の様な知識獲得および知識検証機能を総合的に支援する知識ベース構築支援システムの一環として、診断・解析問題を対象とし、仮説推論の考え方を応用して知識間の整合性維持を自動的に行う機能の実現を試みた知識検証支援システムKNOVについて述べる[21,22]。

知識検証の問題は、不完全かつ矛盾を含む知識ベースを、完全で整合性のある知識ベースに修正、整理して行き、問題解決を正しく行える様にする問題である。知識ベースに含まれる矛盾ルール、不完全ルールとそれを解消するためのルール修正案を仮説と考え、仮説推論を用いて、矛盾解消戦略に基づく仮説生成とその検証を行う事により、知識ベースを洗練化し整合性のあるものにして行く。知識ベースの矛盾検出・修正を支援する機能については、仮説集合に基づいた知識の一貫性維持を行うATMSを利用した知識管理機構として実現した。KNOVの検証機能については、電力系統故障診断および計算機システム復旧支援エキスパートシステムに適用評価して、その有効性を確認した。

2．知識検証

知識検証には大きく分けて2つのレベルがある。1つは、対象問題における診断・解析などの目的（要求仕様）を満たす様に知識の調整をする事(Verification)であり、他の1つは、対象領域における知識そのものの妥当性をチェックする事(Validation)である。例えば電力系統故障診断では、前者は、故障診断において判断すべき事故状況を列挙し、これらを正確に判断できる様に知識を修正する事である。後者の場合は、個々の知識に対して電気法則、系統状態の判断などに照らして対象領域における知識の妥当性を判断する事であるが、一般には妥当性の判断の基準を何に求めるかが難しい。

また、知識検証には静的解析と動的解析の2種類の方法がある。静的解析での知識ベースの矛盾・不完全性の検出は、知識ベースの記述形式を解析し個々の知識相互の関係から、ルール相互の論理的矛盾を見いだす方法である。例えば、if (A) then (B)とif (A) then(not B) の両方が含まれる様な場合は、矛盾と判断される[11]。

しかし、この様な静的解析による検証方法では、推論の文脈に依存した意味的な誤りを検出する事はできない。このためには、実際に知識ベースを推論させて矛盾を調べる、動的解析（動的検証）が必要となる。

知識検証の動的解析の過程は、一般にFig.1 の様に示される。

(1)テスト戦略の設定

どういう（推論）状況を試せば、知識の正しさを検証できるかというテスト戦略を専門家が考える。

(2)動的テスト

テスト戦略により設定された状況下で、検証対象知識を用いた推論を行う。この後の知識の正しさの判定のために、推論内容を蓄えておく必要がある。

(3)知識の正しさの判定

テストケースに対する推論状況を解析し、知識に誤りがあるかどうか調べ、誤りがある場合には、その原因を同定する。

(4)知識の調整

知識の誤りを修正する。修正の方針としては一般化・特殊化、抽象化・具体化等がある。

これらの各過程で、どの様に行えば完全に検証ができるか、またどの様な手順で行えば効率良く検証ができるか、についての知識検証戦略が必要とされる。これは具体的には、テストケースの集合と、テスト手順として与えられるが、一般には上記各過程の途中で状況に応じて決めていく場合が多い。

以上に述べた動的検証において必要となる情報がいくつかある。これを示したのがFig.2 である。知識ベースに含まれる矛盾を検出するためには何が矛盾であるかを判定する基準が必要である。この代表的なものがテストケースである。矛盾が検出された後のルール修正候補の生成には、対象問題領域の知識構造や新たなルールを作成する為の情報が必要である。通常これらの知識は、問題解決の知識と同様に、専門家の頭の中にあり無意識に利用しているものであり、その内容や知識表現について不明な点や曖昧な点が多い。今回の研究ではこの点について検討し、基本的な知識表現形式を定める事を試みた。

この結果、矛盾知識や不完全知識を含む知識集合と、それを解消するための知識修正案を仮説と考える事により、仮説推論を用いて矛盾解消戦略に基づく仮説生成とその検証を行ない、知識ベースを洗練化し整合性のあるものにして行く事が可能となる事が明らかになった。

3．診断問題

知識検証システムを考えるとき、「システムが検証対象とする領域の知識の特徴をどう処理してゆくか」という事は、必ず考慮しなくてはならない点である。特に対象を特定せず検証処理を行おうとすると、それは、知識記述の構文レベルでの検証に留まらざるを得ないが、対象を限定すれば、意味的レベルの検証まで可能となる。そこで、対象を診断問題に限定して検証対象となる知識の範囲を明確にした上で問題を整理し、検証問題を検討した。

我々は、診断問題として、工業プラントを対象とした診断を問題領域として選定した。その知識構造を解析すると、システムの機能の診断は、論理関係のルール化となり、これは、ジェネリックタスクの考え方における状態抽象化(state abstraction) に相当する。このタスクの知識は、要素機能がシステム全体の機能とどの様に関係するか、また構成要素の状態変化がシステムの機能にどの様に影響するかを表現する知識であり、知識の構造はシステム／サブシステムの構成を反映している[10]。

本システムでは、診断知識の構造を表現する手段として、知識フレームの考え方を導入した[21]。知識フレームは対象領域の診断知識の構造を抽出した知識枠組みであり、以下に示す３階層で構成される。

(1)ルールフレーム

一般的にはif-then形式で表現される診断ルールを分類・形式化したものであり、対象領域の問題解決の枠組みと、知識記述形式の枠組みの規定からなる

(2)基本概念

診断ルールを記述するための基本要素（語彙）であり、診断対象プラントの運用状態、状態変化、仮説（原因）などの集合

(3)データフレーム

診断対象の設備機器構成を定義する枠組みであり、構成要素である機器の属性定義、構成要素間の結合関係とその制約定義形式

Fig.3 は問題領域の知識構造と知識フレームの対応を示す例である。

知識検証操作の中では、この知識構造（知識フレーム）を用いて以下の操作を行う。

- 知識構造に基づいた矛盾知識の検出（対象知識ベースの静的解析）
- 知識構造に従った知識の組み替え・生成による知識の自動修正

また、この知識フレームは、以下に示す様に知識獲得の手段としても用いる事ができる。

- 対象問題の知識構造に基づいた、類似問題での知識獲得支援
- 知識概念および階層間の整合性に基づく知識洗練化

4．知識検証の概念モデル

我々は、以下の3つの特徴を持つ知識検証モデルを検討し、その試作及び評価を行った。

(1)仮説推論による動的検証

(2)問題領域に応じた検証メタ知識の定義と利用

(3)ATMSを用いた知識整合性管理

Fig.4 にこの知識検証モデルを示す。Fig.4に示す知識検証モデルは、仮説推論を行う問題解決機構と、ATMSを用いて知識整合性を管理する知識管理機構で構成される。このモデルは診断推論機構を含んでおり、検証を行うために診断推論の実行を制御・管理するメタシステムの構成をとっている。

このモデルでは、検証の対象となる知識を仮説と考える。この仮説に対して専門家の正しい診断をテストケースという形の事実として与え、推論結果と事実から仮説の真偽を調べる。矛盾仮説があった場合には、事実に合う様な新たな仮説（知識）を生成する。最終的に"真"である事が確かめられた仮説集合を、目的とする、無矛盾な知識集合とする。

4．1　知識検証アーキテクチャ

我々の知識検証モデルでは、与えたテストケースを満足する様なルールを生成する事がゴールである。

仮説検証では、与えられた初期知識あるいは知識修正案を仮説とし、この仮説に矛盾があるか否かを検証する。矛盾の検出においては、知識ベースを構成する各知識階層に対応するテストケースを与えて診断推論を行い、その結果生じる矛盾を矛盾検出知識を用いて抽出し矛盾仮説を特定する。

仮説生成では、診断結果とテストケースが異なる場合に、テストケースに合う様に、矛盾仮説（ルール）の結論部・条件部を組み替えて新たな仮説を作り、それを修正案とする。この仮説生成に対し、生成の制約条件としての知識フレームと生成の手がかりとなる仮説生成知識を与える。これらの情報は、対象とする問題領域に固有に存在する基本的な概念や法則を反映したものであり、専門家より与えられる。

4．2　矛盾知識管理へのＡＴＭＳ適用

本システムで扱う診断知識は、ルール・基本概念・データの階層関係で定義され、その内容に応じてモジュール化される。個々のモジュールに含まれるルール群は、ルールを構成する条件・結果の各項を要素とするAND/OR回路と考える事ができる。従って、ルールの矛盾／不完全構成要素の検出は多重故障診断の一種と考えられ、矛盾知識の管理にATMS手法が利用できる[14,15]。

知識検証の場合には、仮説推論の要素を以下の様に考える。

(1)対象　　：診断知識
　観測事実：故障時の現象と推論結果
(2)モデル　：専門家の知識・判断
　挙動予測：テストケース
(3)対象とモデルの相違：
　　推論結果（観測事実）と
　　テストケース（挙動予測）との違い
　故障候補：診断知識要素（ルール、基本概念など）

従って、ATMSを用いて矛盾知識を管理するために、前提に対して既知の事実および知識、仮説に対して検証対象とする正しさの確認されていない診断ルールを対応づける。他のデータについてはATMSの定義通りであるが、それらの意味づけは以下に示す通りである。

・ノード　a.前提..テストケース、矛盾検出知識
　　b.仮説..診断知識要素（診断ルールあるいは基本概念の集合）
　　c.仮説および前提からの導出
・理由付け　〈導出ノード群〉　または　導出ノード
　　推論の基本単位
・環境　仮説の組み合せよりなる集合
・ラベル　環境からなる集合で、各ノードが持ち、そのノードが従属する仮説を示す。
　　即ち推論結果を導出したルールあるいは基本概念の集合。
・コンテクスト
　　無矛盾な環境の仮説とそれらの仮説から導出されたノードの集合。
　　即ち正しい推論結果を導出するルールあるいは基本概念と、それらからの推論過程を示すノードの全て。

この考えに基づいた矛盾知識管理方式をFig.5 に示す。ここでは診断ルールを構成する個々の基本概念A1,Zを仮説としているが、診断ルールを仮説の単位としても同様である。仮説A1は診断ルールの条件節となっており、仮説Zは診断ルールの結論節である。また、知識検証を行うためのテストデータa1,a2,d1,d2,d3と矛盾検出知識Cを事

実（前提）として与えている。

ルールの推論実行に応じて、ルールを構成する基本概念の実行順序（導出）関係から、基本概念とそれぞれの推論時の変数の入出力関係に基づいて、ルールの実行単位をノードとする推論ネットワークが作られる。この時、導出ノードにはその根拠となる仮説集合が環境として保存される。Fig.5では、A1を基本概念として持つ診断ルールの、入力データがa1,a2 である2つの推論実行に対して、2つのノードが作られる。推論の結果はそれぞれd1,d2 となるが、何れのノードの環境も{A1,Z}となる。

仮説検証（矛盾検出）時には、矛盾検出知識により推論結果をチェックする。矛盾があれば、該当する仮説集合をATMSのnogoodデータベースに登録し、推論ネットワークのノードの上位／下位の関係から、ネットワーク内の関係するノードを矛盾とする。Fig.5では、推論の結果であるd2を入力とした矛盾検出知識が実行され矛盾となる。その結果、理由付けとして登録されている導出ノード(*)が矛盾とされ、それに関係した仮説が上位／下位の関係より矛盾とされる。Fig.5の場合には、矛盾とされた環境{A1,Z}がnogoodデータベースに登録される。これは基本概念A1,Zで構成される診断ルールが矛盾である事を示している。

Fig.4に示した知識検証モデルにおいて、知識管理機構は成功仮説・失敗仮説(矛盾知識)と、それらの導出関係（推論過程）をATMSにより管理する。問題解決機構では、ATMSによる管理情報を用いる事により、仮説生成や推論の効率向上を図っている。仮説生成時には、すでに矛盾である事が判明している仮説を含む新たな仮説は作成しない様に制御しており、仮説推論時には、登録されている導出関係を参照して過去に失敗した推論は行わない様にして無駄な推論を排除している。一般的に、生成される仮説数とその検証（推論）回数は知識の複雑さに対して指数的に増大するが、ATMSを利用する事によって、これを効率的に制御・管理する事ができる。実際の診断問題への適用を考えるとき、推論は複数の知識モジュールにわたり多段に行われ複雑となる場合が充分考えられるが、その様な状況に対し、ATMSの利用が、矛盾知識の検出・知識整合性の管理・処理の高速化に有効性を発揮する。

４．３　検証メタ知識

本システムにおいては、検証操作のためのメタ知識として、テストケース、矛盾検出知識、知識フレーム、仮説生成知識を用意した。これらは、検証処理から独立させ、各々の内容に応じた知識表現形式を与えて定式化した。前者2つは仮説検証用、後者2つは仮説生成用メタ知識であり、現状では、何れも専門家が与え、その妥当性を判断するものである。以下にこれらの概要、定義内容、抽出方法等について述べる。これらの検証メタ知識の具体例をFig.8に示す。

４．３．１　仮説検証メタ知識

(1)テストケース

テストケースとは、動的解析における規範である。テストケースとして定義すべき項目は、次に述べる3つである。

・診断対象システムの構成と状態（データベース）
・故障発生時の現象・状況（診断システムへの入力）
・故障原因（診断システムの出力であり、診断結果の正否の判定基準）

テストケースとして抽出すべき内容の必要十分性は、診断の目的や度合（どの程度まで詳しく診断・解析を行うか）に依存し、検証対象知識ベースに含まれる個々の知識がそれぞれの目的や機能を満足するか否かを基準としてテスト仕様が決定される。この様なテストケースが持つべき性質には次の３項目がある。

・正確さ　..個々のテストケースの正しさ
・完全性　..テストケースの集合が検証対象知識のカバーする診断領域全体を包括する事
・無矛盾性..テストケースが相互に矛盾しない事

特に完全性、無矛盾性を保証する意味で、テストケース相互の機能的な階層関係、即ちテストケースの構造性を考慮をすれば、テスト戦略を考える場合に有効である。テストケースの抽出と、それが上記の性質を満たすかどうかの判断は専門家が行う。従って、テストケースを抽出しようとする場合、まず、診断対象システムにおいて想定し得る診断結果を導出するすべての故障を網羅し、それらの故障パターンを分類し、抽象化して、代表的な故障ケースを選択する事になる。

(2)矛盾検出知識

矛盾検出知識とは、検証対象知識が誤った知識を含んでいないかどうかを、診断の結果あるいはその診断過程より判断するものである。矛盾検出知識では次の定義を行う。

・矛盾（誤っている）と判断する根拠

矛盾検出知識は、診断結果はこうあるべきであるという規範と実際の診断結果とを比較し異なる場合は何れかの知識に誤りがあると判断するものである。ここで言う規範とは、(1)で述べたテストケースとして定義される個々の事実と、それらを抽象化した診断対象とする問題領域に依存する知識がある。

矛盾検出知識の内容は、テストケースとの比較を除いては、診断対象とする問題領域に依存するものがほとんどである。矛盾検出知識抽出の具体的方針としては以下に示すものがある。

・診断目的と診断結果が矛盾する（妥当ではない）
・診断の目的に応じた診断結果ではない
（的はずれである）
・システムの持つ状態や性質と診断結果が矛盾する
（原理を逸する判断）
・複数の診断結果が存在する場合、それらに相互矛盾がある

矛盾検出知識は、検証対象知識（診断知識）を用いた診断結果を、誤り検出の判断材料としている。従って、矛盾検出知識の内容は、検証対象知識に記載されている診断結果の表現方式や詳細度、さらに診断結果を得るために参照する問題領域のデータベースの表現方法・内容に依存する。従って矛盾検出知識の表現形式は、診断知識の表現形式と同じにする事が可能であり、診断知識に対して親和性の高い矛盾検出知識を定義できる。

(3)矛盾検出

矛盾知識管理へのATMS適用で述べた様に、検出された知識は一般的には複数の仮説組み合わせとなる。この中でどの知識要素が真に矛盾しているかを判断するためには、より分解能のすぐれたテストケース、矛盾検出知識を必要十分なだけ用意する必要がある。またさらに、矛盾検出の為の推論をできるだけ小さな単位、例えば診断知識の推論毎に行う事も必要である。個々の矛盾を分離するための有効なテストケース集合、矛盾検出知識を決める事は専門家の高度な判断を必要とする。

４．３．２　仮説生成メタ知識

(1)知識フレーム

通常、問題解決知識は対象とする問題領域に応じて知識表現に制約を持たせ、定められた枠組みに添って知識を記述している。Fig.3に示した様に、知識フレームは診断知識の記述内容の規則性を知識の枠組みとして定義したものである。本システムでは、専門家が診断知識の条件記述や結果記述部分の記述内容の規則性を見出し抽出する。その定義は、知識モジュール毎にルールに記述すべき基本概念の範疇を、記述すべき順に並べたものとしている。

(2)仮説生成知識

仮説生成知識とは検証対象診断システムが目的とする問題領域の定義であり、仮説検証機能で誤っているとみなされた知識を修正する際に用いる情報である。

仮説生成知識は対象領域の概念を階層構造として定義したものであり、そこに定義された概念間の関係は基本的に、「抽象化」，「具体化」，「並列概念」の3つである。また更に、診断知識の種類や検出した矛盾の内容に応じて仮説生成の範囲を制限する仮説生成制約も併せて定義し、無意味な仮説の生成を抑制する。

(3)仮説生成

仮説生成は仮説生成知識を用いて検出された矛盾知識の修正案を作成し新たな仮説とする。仮説生成知識を参照しながら次の3つの戦略により仮説を生成する。

・抽象化..仮説の記述内容の抽象化・一般化
ルールの条件記述、結論記述の概念を上位概念へ変換、複数の具体的ルールから１つの抽象的ルールへの集約
・具体化..仮説の記述内容の具体化・詳細化
ルールの条件記述、結論記述の概念を下位概念へ変換、１つの抽象的ルールから状況に合う複数の具体的ルールへの展開
・並列代替案..並列概念による記述内容の置き換え
ルールの条件記述、結論記述の概念の異種同位概念との置き換え

これらの戦略を用いて矛盾知識に対応する仮説を作成し、テストケース、矛盾検出知識を満足する様なルール修正案を見出す。複数の矛盾知識が検出された場合には、そ

れらの矛盾知識の全てについて仮説生成を行う。仮説生成における知識フレームの役割りは、仮説生成知識を参照して代替概念を生成する際の、概念の妥当性の制約および保証であり、矛盾知識上の当該修正概念について、仮説生成知識上代替概念となり得る範疇を規定している。

診断知識の欠落のために正しい判断ができない場合、即ち、診断推論が行われ（ルールが発火）され何らかの誤った結論を出力した場合には、仮説生成知識（概念樹）をもとに上位／下位／並列概念から欠落したルールを生成する。知識検証を行う場合にはある程度の初期知識が既に作成されている事を想定している。従って初期知識におけるルールの部分的欠落や抽象レベルの不適切さを、全てのテストケースおよび矛盾検出知識を満足する様に、仮説生成知識（概念樹）を利用して、欠落ルールの補正あるは不適切な抽象レベルの調整（抽象化／具体化）を行い新たなルールを作成する事が可能である。

仮説生成の例をFig.6 に示す。図では、元のルールの条件部の基本条件"at the side of the line(L)"に注目し、具体化の戦略を用いて仮説生成知識を参照し、概念sideを詳細化した2つの概念"supply side","receive side"による修正案（仮説）を作成している。

(4)生成仮説の検証

抽象化、具体化による代替えルール候補は知識フレームおよび仮説生成制約の制約のもとで、可能性のあるもの全てを作成し、それらの中ですべてのテストケースおよび矛盾検出知識を満足するものだけを選ぶ様にしている。即ち、基本的には「Generate&Test」であり、対象とする問題領域の知識を反映させた制約により生成する仮説（代替えルール候補）の数をおさえる様にしている。

複数の矛盾ルールを修正する場合には、それぞれの矛盾ルールに対して、ルール修正候補（仮説）の生成・検証を行う。生成された仮説を検証する時には、矛盾検出時と同様に個々の仮説に対してテストケースを用いて推論を実行し推論ネットワークに推論履歴を成功、失敗を含めて追加して行く。このため複数の矛盾ルールを修正してもそれらを修正するために生成された仮説の推論結果が推論ネットワークに（ＡＴＭＳのノードの環境およびnogoodデータベースに）整合性を保たれた状態で蓄積されていき、矛盾無く処理が行われる。

この様に新しく作成されたルールは、与えられた要求仕様（テストケース集合、矛盾検出知識）を満足するため、要求仕様が適切に与えられる限り過度にルールを拡大する事や相互に矛盾する事は無い。

５．知識検証支援システム：ＫＮＯＶ

５．１　システム構成

第2章から第4章で述べた知識検証方式を、Prologの拡張言語であるExtended Self-contained Prolog(ESP)[16]を用いてPersonal Sequential Inference machine-II (PSI-II)上に、知識検証支援システム:Knowledge Verification system(KNOV)として実現した。KNOVは、Fig.7に示す構成となっている。検証の動きを司る問題解決機構を、診断推論・矛盾検出・仮説生成の3つの機能に分割し、各々の検証用メタ知識と対応をとりながら検証対象に応じた柔軟な検証操作ができる形としている。知識管理機構は推論ネットワーク・矛盾知識データを保持し、それらの情報管理機能を持っている。矛盾知識の管理は基本ATMSを利用している[17]。今回実現した範囲では、矛盾検出に関しては個々に独立した矛盾の検出を対象とし、診断ルールの推論毎に矛盾検出・同定を行っており、複合矛盾の検出は対象としていない。

５．２　知識検証例

電力系統故障診断[18,19]、計算機システム復旧支援[20]の2つの実用化されているエキスパートシステムに適用し、KNOVの機能検証および評価を行った。本論文では前者の適用・評価の例を紹介する。

５．２．１　対象問題

電力系統とは、電気を発電機から負荷まで運搬するための設備であり、送電設備や工場などの構内受配電設備がこれに相当する。電力系統はライン・母線・変圧器・調相設備等が開閉設備を介して接続するもので、これら開閉設備には、系統用保護リレーが設置されており、系統設備に故障が発生したときリレーが異常を検知し開閉設備をしゃ断する。これにより故障設備が健全系統より切離され、事故の波及を防ぐ。電力系統故障診断は、系統で故障が発生し、リレーが動作して開閉設備がしゃ断し、系統上の何れかに停電箇所ができた状況から事故設

備（区間）と事故原因（種類）を診断するものである。

５．２．２　検証例と評価

電力系統故障診断の場合の診断知識と、診断知識の検証操作に必要な検証メタ知識の例をFig.8 に示す。

検証対象となる電力系統故障診断用の診断ルールを(a)に示す。これは知識モジュールrule1に含まれるルールであり、「動作したリレーの種類が78sでありそれにより母線BUS側の遮断器が遮断したならば事故種類は短絡であり事故区間はその母線BUSである」という知識を示している。

(b)は(a)のルールの正否を判定するテストケースで、異った事故状況の各々に正しい判断を下せるかを試すものである。例えば(b)-2は「動作したリレーが50s,78sでそれにより遮断器sw5,sw6が遮断した場合、bc_line_1Lが短絡事故である」と判断すべきである事を示している。(c)は矛盾検出知識であり知識モジュール単位に定義する。(c)-1は「診断結果がテストケースで定義した事故区間・事故種類と異なればルールに誤りがある」、(c)-2は「事故区間が停電していなければルールに誤りがある」事を示しており何れも知識モジュールrule1に対する矛盾検出知識である。

(d)は知識フレームと仮説生成制約を示したものである。ここに示した知識フレームは、「リレーの動作原理から事故の場所および原因の候補を求める」ための知識モジュールrule1に対応するものであり述語frameで定義されている。この知識フレームは入力変数がリレーを示すRY、出力変数が事故の場所、原因を示すAREA,KINDであり、ルール条件部は基本概念群activated_relay,facility_side,system_state、結論部は基本概念群faulty_kind、faulty_areaで構成される事を示している。また述語flagのprimitive_groupは基本概念群と仮説生成知識との対応関係を定義している。(d)に示す述語flagのstrategyとerrorが制約知識の例である。strategyはルール修正案（仮説）を生成する時の基本概念選択の範囲を規定する指定である。例えばfacility_side,system_stateのpart_ofの指定は仮説生成知識の上位・下位１階層および兄弟関係の中から選択する事を示している。またfaulty_kind,faulty_areaのallの指定については対応する仮説生成知識の全ての範囲から選択する事を示している。errorは検出した矛盾に応じて基本概念選択の範囲を規定する指定である。ここに示した例では矛盾検出知識2002で矛盾を検出した時には、ルール条件部のfacility_sideとルール結論部のfaulty_kindの部分だけを組み替えてルール修正案（仮説）を生成する事を示している。(e)は仮説生成知識の一部を示したものであり、対象とする問題領域である電力系統の構成、運用等に関する概念の階層構成を定義したものである。仮説生成は、検出したルールおよび矛盾の内容に応じてこの仮説生成知識を、知識フレームおよび仮説生成制約の範囲内で辿り、対応する基本概念を選択しそれらを組み合わせる事により行う。

Fig.9 はKNOVのマンマシンインタフェースでありKNOVを用いた知識検証操作を示している。(a)はメニュー・対話画面を示している。この画面により検証対象の診断知識、使用する検証メタ知識、検証過程の表示方式の指定および検証結果、生成仮説、ATMS内部データ等の表示を行う。(e)は検証過程で検出された知識ベースの中の矛盾知識を示している。ここではFig.8(a)のルールに矛盾がある事が示されている。(d) は(e)で示した知識を誤りであると判断した矛盾検出知識であり、Fig.8(c)-1の矛盾検出知識を示している。(c)は矛盾知識(e)に対して生成・検証された最終的な修正知識を示している。矛盾知識は、Fig.8(d)の仮説生成知識のbusの概念の並列概念であるlineを用いて修正され、基本概念facility_side,faulty_areaのパラメータがそれぞれbusからlineに変えられた修正知識が生成された事がわかる。更に、(b)は最終的に修正された知識ベースによる診断結果を示しており、Fig.8(b),(c)に示したテストケースおよび矛盾検出知識を満足する、正しい診断を行う事を確認する事ができる。

知識検証の評価は以上に示した例を含んだTable 1(a)に示す診断知識と検証メタ知識に対して行った。またその評価結果をTable 1(b)に示す。評価ケース1，2はルールの詳細化、評価ケース3はルールの一般化、評価ケース4は並列概念によるルールの置き換えのそれぞれに対する評価を行ったものである。何れの場合にも矛盾知識を正しく検出し、それが矛盾の無い様に適切に修正される事を確認した。

それぞれの評価ケースにおいて検出されるべき矛盾知識は一つであり、それに対応して生成された仮説の数はTable 1(b)の仮説数に示した通りである。

a.検出した矛盾知識に対応して生成された仮説

（ルール修正案）の総数

b.仮説生成において知識管理機構により矛盾が検出された仮説、即ち過去の推論結果から矛盾ルールである事がわかった仮説の数

c.推論を実行して矛盾である事がわかった仮説の数

d.最終的にテストケースを満足する無矛盾な仮説（最終修正ルール）の数

知識検証の処理にかかる時間はこれらの何れの評価ケースの場合にも1分以内であり実際の使用においても問題は無いと考えられる。

6．結論

我々は知識検証方式についてその基本的な考え方を明らかにした。さらにそれに基づいた知識検証支援システムKNOVを試作し検証方式の実証・評価を行いその有効性を示した。本知識検証方式の有効性は以下に示す3点にまとめる事が出来る。

(1)矛盾検出・仮説生成の自動化による省力化

矛盾検出、仮説生成メタ知識を与えれば、システムが自動的に知識修正案の全ての可能性をチェックし正しい修正案を導く。この為、人間が検証を行う場合の様に、先入観による思い込みや不注意によるチェック漏れは生じないし、時間の節約も可能である。

(2)テストケース群によるテスト

ATMSにより矛盾知識を集合として管理するので、複数のテストケースによる知識検証が機械的にできる。人間の場合、テスト戦略を立てて逐次的に矛盾部分の検出・修正・確認を行うが、本方式では複数の修正案を集合として同時に検証できるので、「ある状況では正しいルールが、別の状況では誤った診断を行う」様なケースも矛盾なく管理ができる。

(3)検証用メタ知識の定義

知識フレーム・矛盾検出知識・仮説生成知識を、検証処理手続きから独立させ、対象とする問題領域毎に設定できる構成となっており、問題領域に応じた知識検証システムとする事ができる。

本論文で述べた検証方式およびその実現により知識検証の自動化への道が開かれたと考えている。今後は実用に耐えるシステムに発展させたい。そのためには以下に示した課題をさらに検討する必要がある。

(1)検証メタ知識

検証メタ知識は、知識検証、さらには知識獲得にも用いられ得るものであり、一般的な診断知識よりは高度な内容を持つ。従って、それらの抽出・整理・定式化をどの様にすればよいか、また、それらの知識の完全性や十分性をいかに定義し確認するか、を明らかにする必要がある。

(2)矛盾知識の同定

相互に関連する複合矛盾知識を同定するためには矛盾検出戦略またはアルゴリズムの開発が必要である。更に、それを効率良く行うためのテストケース、矛盾検出知識の設定方法を明らかにする必要がある。

(3)仮説生成の多様化

対象問題領域の概念の階層構造を用いた仮説生成方式以外に、ルールを形成する基本概念の省略・追加・順序の組み替え等、状況に応じた様々な仮説生成方式を提供できる様にする必要がある。

尚、本研究は第五世代コンピュータプロジェクトの一環として（財）新世代コンピュータ技術開発機構からの委託により行ったものである。多くの貴重な意見と御指導を頂いた第五研究室生駒憲治室長（現NTTデータ通信㈱）および同研究室の皆様に感謝致します。

◇参考文献◇

(1) Hayes-Roth,H.,Waterman,D.,A.,Lenat,D.,B.: Building Expert Systems,Addison-Wesley(1983).

(2) Gaines,B.,R.:An Overview of knowledge-acquisition and transfer,International Journal of Man-Machine Studies,Vol.26, pp453-472(1987).

(3) Davis,R.,Lenat,D.B.:Knowledge-Based Systems in Artificial Intelligence,New York,McGraw-Hill(1982).

(4) Boose,J.:Personal constract theory and the

transfer of human expertise,Proceedings of the National Conference on Artificial Intelligence,Austin,Texas, pp27-32(1984).

(5) Boose,J.H.,Bradshow,J.M.:Expertise transfer and complex problems: using AQUINAS as a knowledge-acquisition workbench for knowledge -based systems,International Journal of Man-Machine Studies,Vol.26,pp3-28(1987).

(6) Kahn,J.,Nowlan,S.,MacDermott,J.:Strategies for Knowledge Acquisition, IEEE transactions of pattern analysis and Machine intelligence, V.PAMI-7,No.5, pp511-522(1985).

(7) Esheiman,L.,MacDermott,J.:MOLE : A Knowledge Acquisition Tool That Uses its Head, Proceedings of the National Conference on Artificial Intelligence,pp950-955(1986).

(8) Klinker,G.,Bentolila,J.,Genetet,S.,Grimes,M., MacDermott,J.:KNACK - Report-driven knowledge acquisition, International Journal of Man-Machine Studies,Vol.26,pp65-79(1987).

(9) MacDermott,J.,Dallemage,G.,Klinker,G., Marques,D.,Tung,D.:Explorations in How to Make Application Programming Easier, Proceedings of the First Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop,pp134-147(1990).

(10) Chandrasekaran,B.:Generic Tasks in Knowledge -Based Reasoning : High-Level Building Blocks for Expert System Design", IEEE Expert, Fall, pp.23-30(1986).

(11) Nguyen,T.,A.,Perkins,W.,A.,Laffey,T.,J., Pecora,D.:Checking an Expert Systems Knowledge Base for Consistency and Completeness,Proceedings of 9th IJCAI, pp375-378(1985).

(12) O'Leary,D., O'Keefe,R.:Verifying and Validating Expert Systems, Tutorial: MP4 of IJCAI-89(1989).

(13) O'Leary,T.,J.,Goul,M.,Moffitt,K.,E., Radwan,A.,E.:Validating Expert Systems, IEEE Expert, June, pp51-58(1990).

(14) De Kleer,J.:An Assumption-Based TMS, Artificial Intelligence, Vol.28, pp.127-162(1986).

(15) De Kleer,J., Williams,B.C.: Diagnosing Multiple Faults, Artificial Intelligence, Vol.32, pp.97-130(1987).

(16) 紺田、近山 :ＥＳＰプログラミング、 TM-134,ICOT(1985).

(17) 太田、井上 :仮説推論システムAPRICOT/0 ユーザーズマニュアル、TM-676, ICOT(1989).

(18) 守口、功刀、石川 :電力系統事故判定・復旧ガイダンスシステム、人工知能学会誌、Vol.3,No.1, pp46-52(1988).

(19) Moriguchi,S.,Tanaka,T.,Ishikawa,K.:Expert Shell for Power Systems Diagnosis,Proceedings of International Workshop on Artificial Intelligence for Industrial Applications, Hitachi City, JAPAN, May, pp.585-590(1988).

(20) Ishii,S.,Abe,T.,Hidai,Y.,Tanaka,A.: An Operations Advisor for On-line Computer Banking System with Graphics Interface, Proceedings of IEA/AIE-89,pp568-576(1989).

(21) 石川、田中、山尾 :知識獲得支援システム、 1989年度人工知能学会（第3回）全国大会論文集、 12-9(1989).

(22) 石川、田中、山尾 :知識検証支援システム、 情報処理学会第40回全国大会講演論文集、 7C-3(1990).

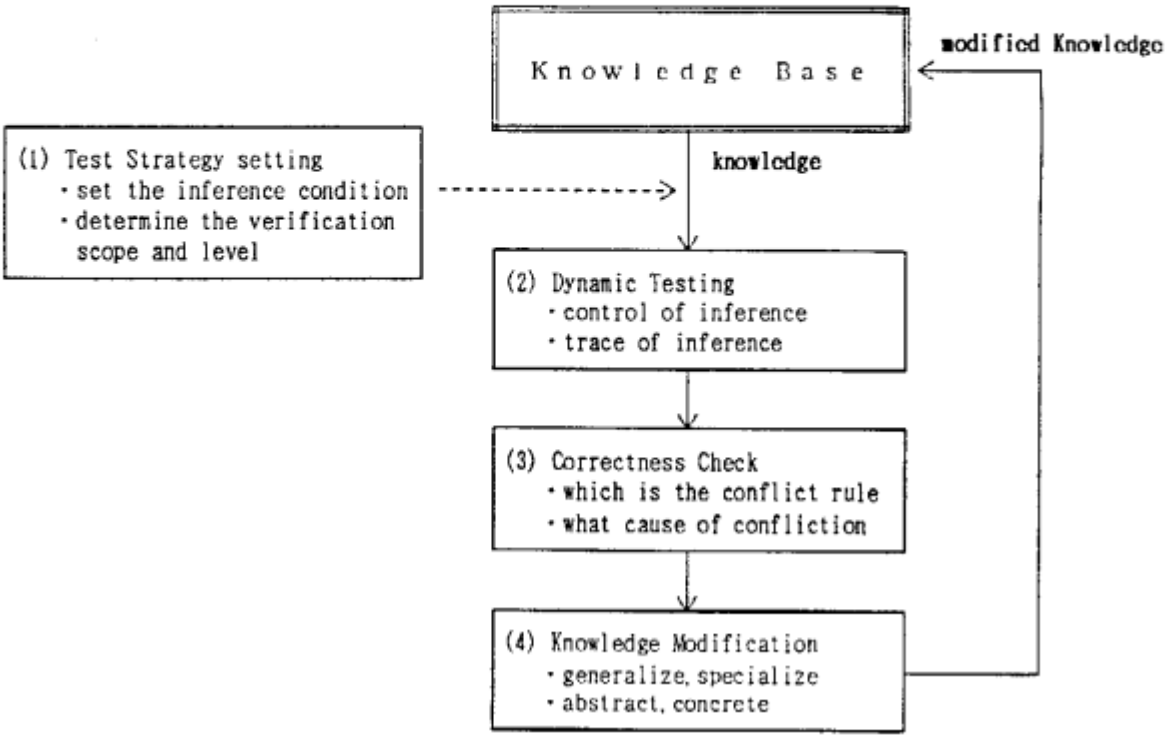


Fig. 1 Operational Flow of Knowledge Verification.

図表-1

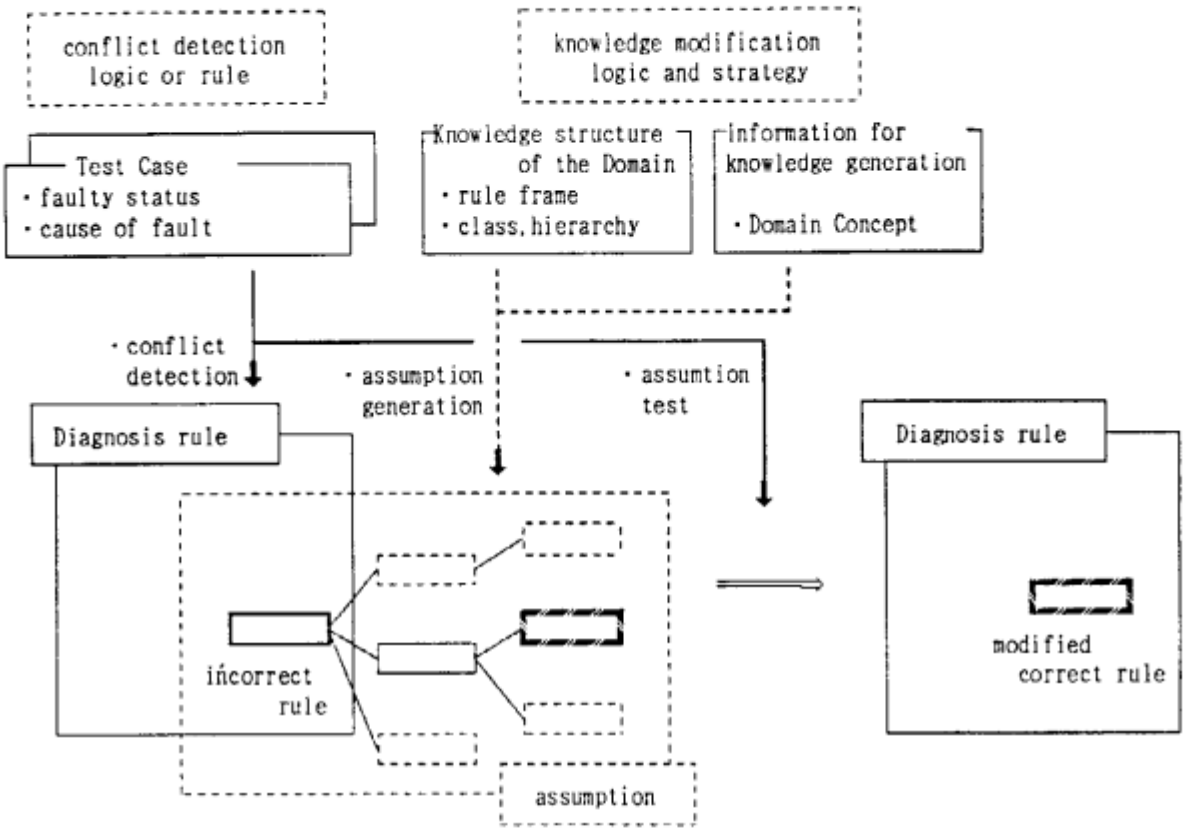


Fig. 2 Information needed for Verification.

図表-2

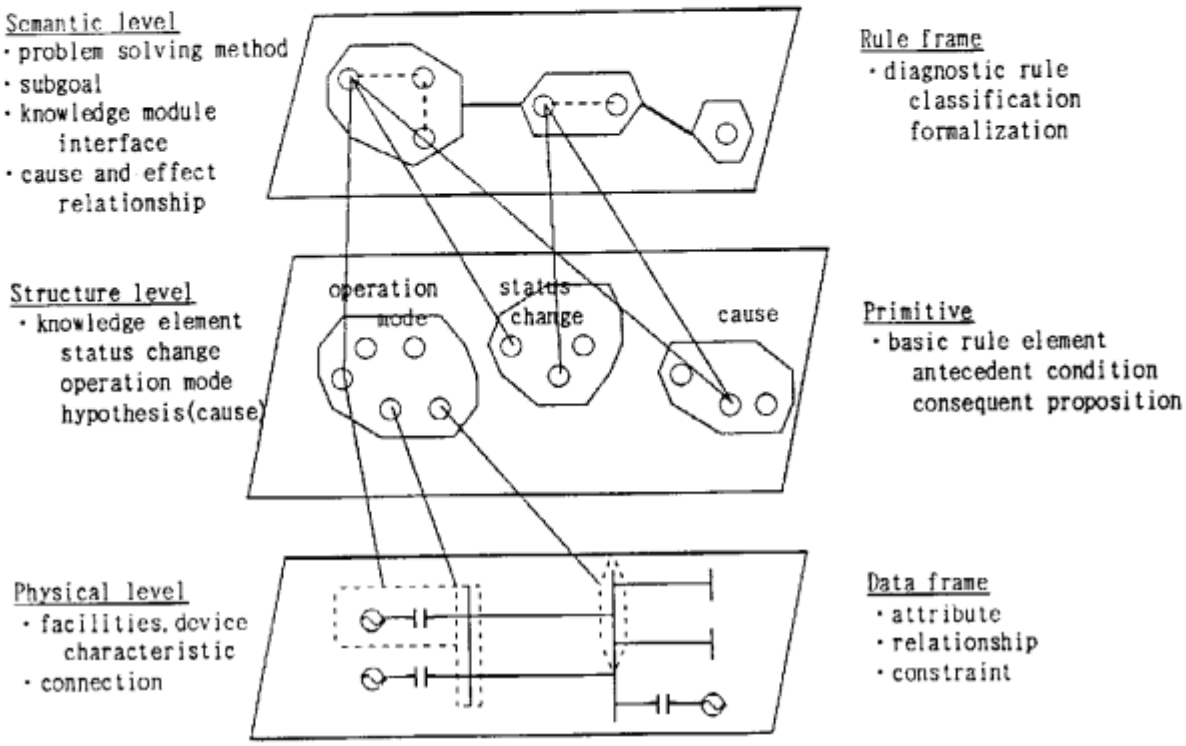


Fig. 3 Knowledge frame for Diagnosis.

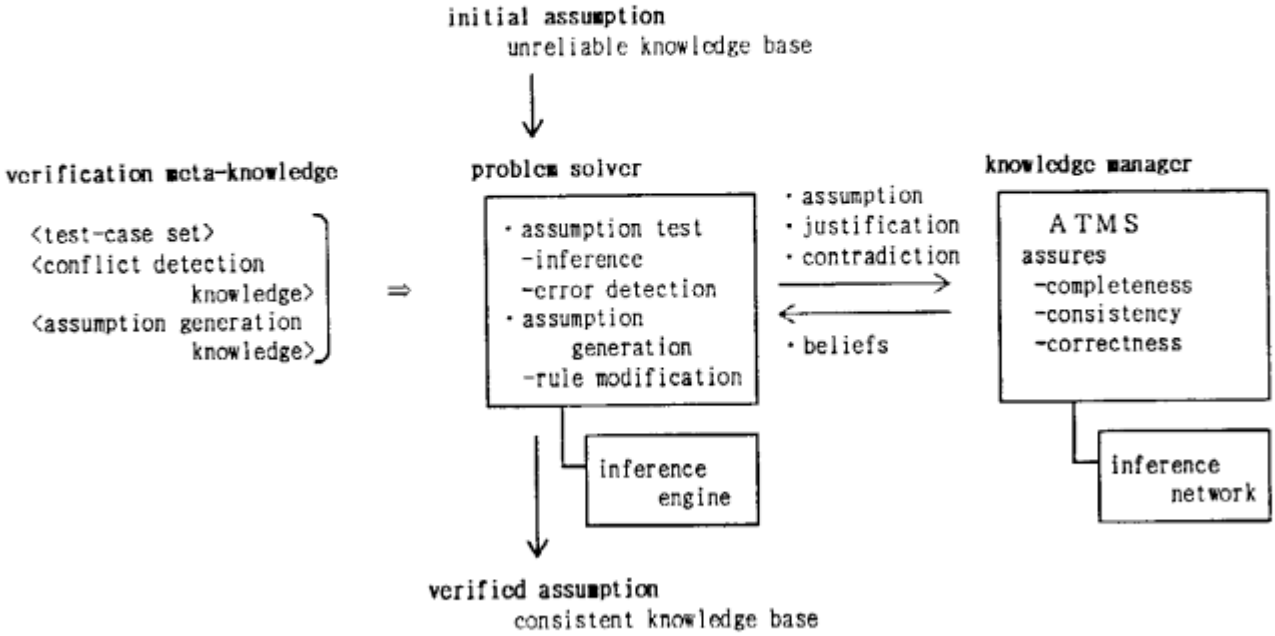


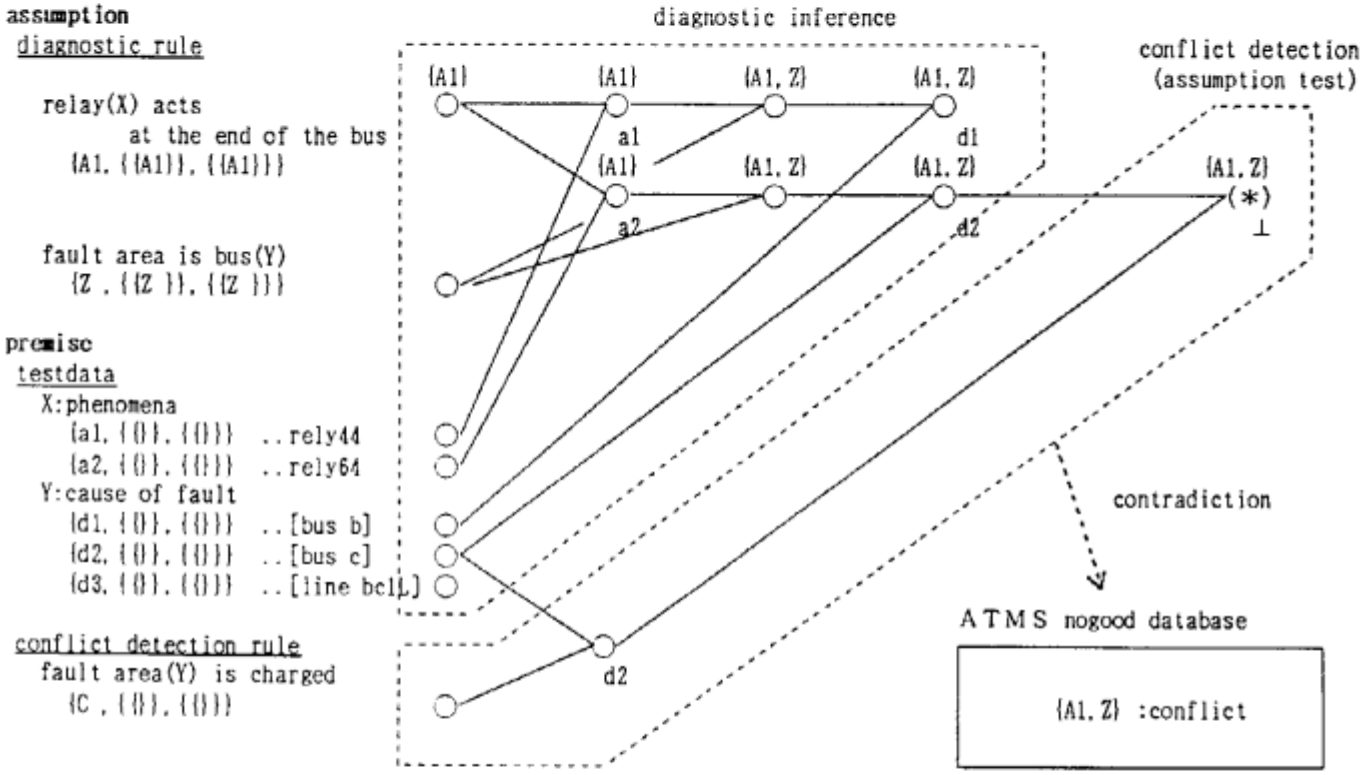
Fig.4 Knowledge Verification Architecture.

図表-4

diagnostic rule : IF relay(X) acts at the end of the bus THEN fault area is bus(Y).

conflict detection rule : IF fault area(Y) is charged THEN contradiction.

(a) rule example



(b) inference network

Fig. 5 Knowledge consistency maintenance based on ATMS.

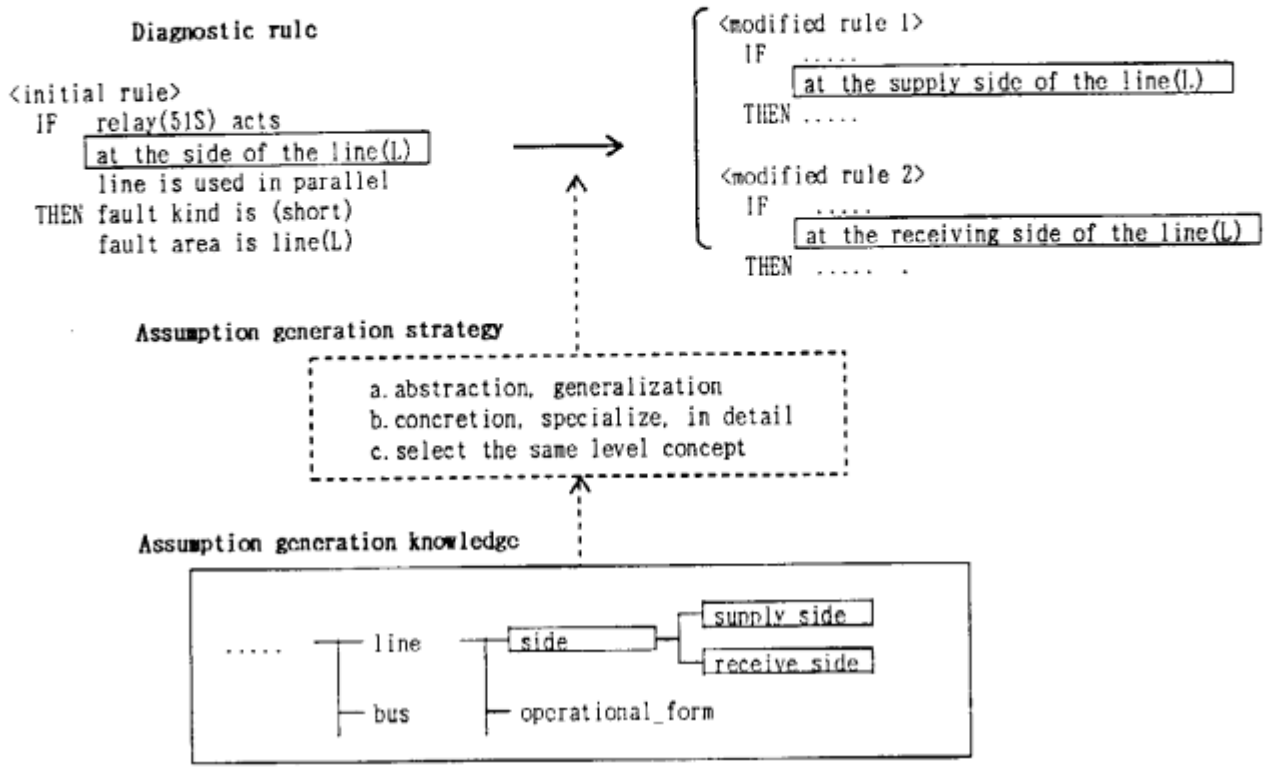


Fig. 6 Example of Assumption generation.

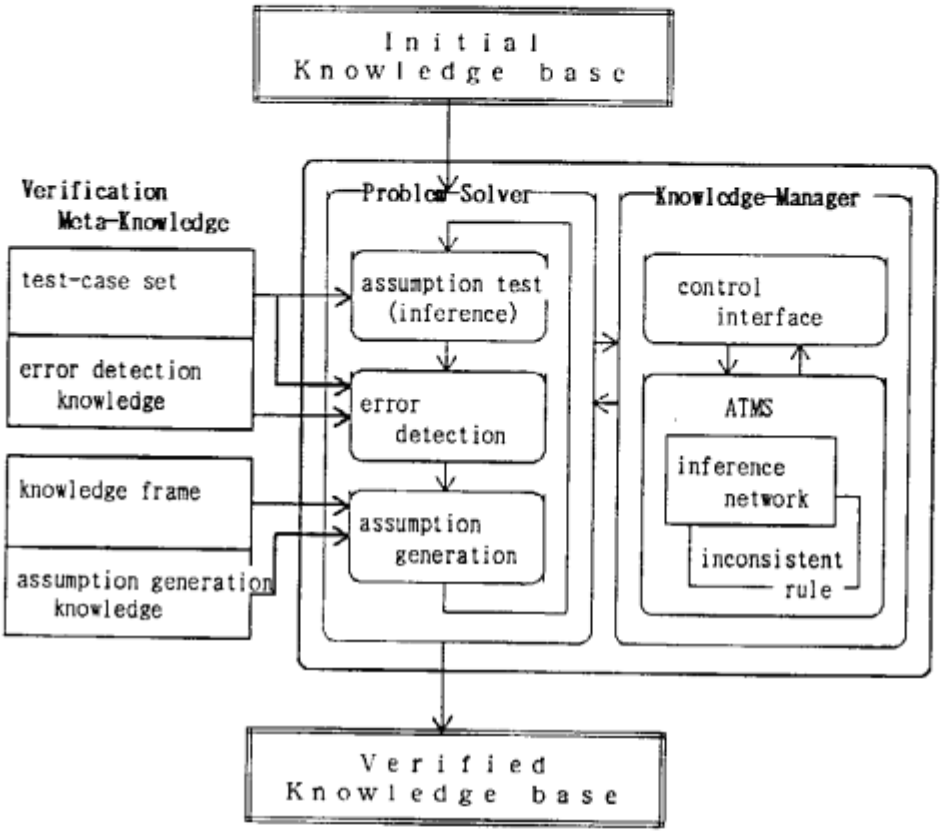
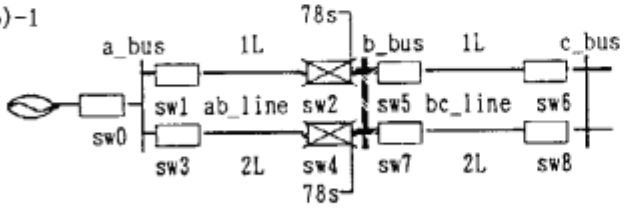


Fig. 7 System Configuration of KNOV.

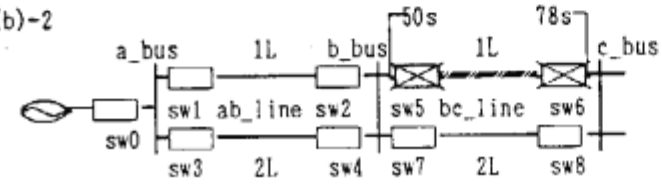
```
rule1(1,24,[RY],[RY,AREA,KIND]):-
        system_state(activated_relay,222,SW,RY,'78s',1),
        facility_side(bus_side,2,BUS,SW),
        faulty_kind(short,11,KIND),
        faulty_area(bus,12,BUS,AREA).
```

(a) Diagnosis Rule

(b)-1



(b)-2



<definition of test case>

test case	(input) relay	(output) area/kind
(b)-1	78s(sw2) 78s(sw4)	b_bus /short
(b)-2	50s(sw5) 78s(sw6)	bc_line_1L /short

(b) Test Cases

(c)-1
```
error1(1,2000,rule1,[RY,AREA,KIND]):-
        system_state(test_case,3,[RY],[TEST_AREA,TEST_KIND]),
        operation(equal,11,KIND,TEST_KIND),
        not(operation(same_factor,31,AREA,TEST_AREA)).
```

(c)-2
```
error1(1,2002,rule1,[RY,AREA,KIND]):-
        not(system_state(all_area_is_off,911,AREA)).
```

(c)-3
```
error1(1,2003,rule1,[RY,AREA,KIND]):-
        relay_kind(short,11,RY),
        faulty_kind(ground,21,KIND)).
```

(c) Error Detection Knowledge

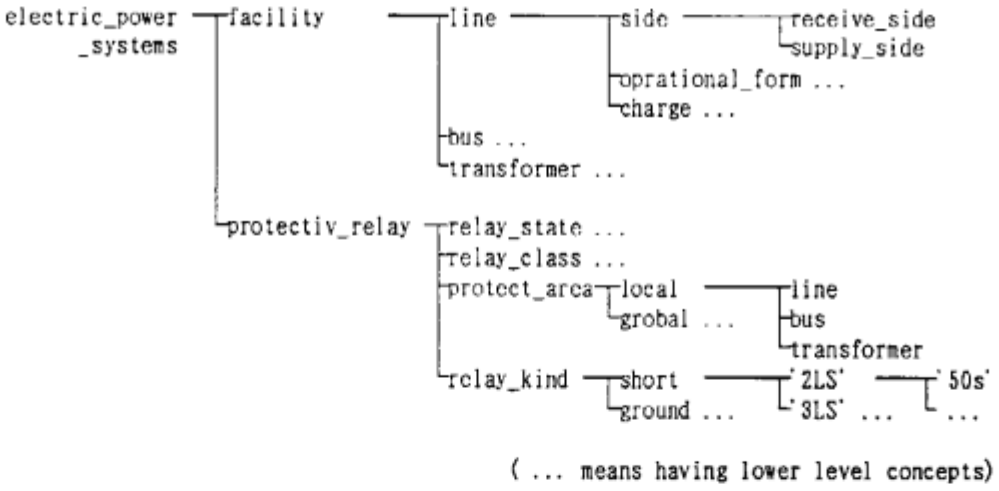
```
frame(rule1, ['RY'], ['AREA', 'KIND'], .
          [[activated_relay, facility_side, system_state], [faulty_kind, faulty_area]]).

        flag(primitive_group, [activated_relay, [relay_kind]]).
        flag(primitive_group, [facility_side, [facility]]).
        flag(primitive_group, [system_state, [operational_form, charge, relay_state... ]]).
        flag(primitive_group, [faulty_kind , [relay_kind]]).
        flag(primitive_group, [faulty_area , [protect_area]]).

        flag(strategy, [facility_side, [part_of]]).
        flag(strategy, [system_state , [part_of]]).
        flag(strategy, [faulty_kind, [all]]).
        flag(strategy, [faulty_area, [all]]).

        flag(error, [2002, [facility_side, faulty_area]]).
        flag(error, [2003, [facility_side, faulty_kind]]).
```

(d) Knowledge Frame and Assumption Generation Constraint



(e) Assumption Generation Knowledge

Fig.8 Example of Knowledge for Electric Power Systems.

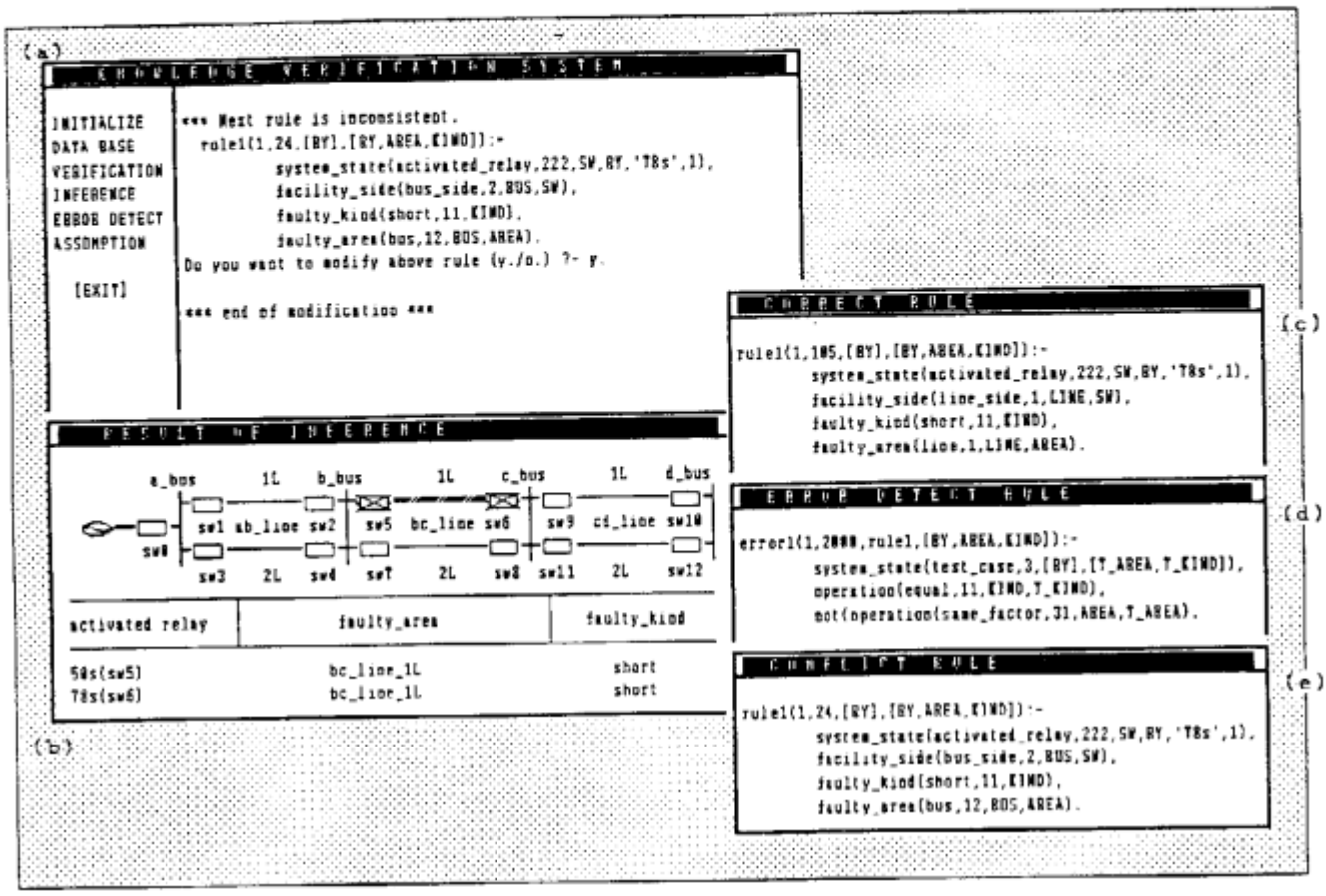


Fig. 9 Man Machine Interface of KNOV.

Table 1 Evaluation of Knowledge Verification.

(a) Diagnostic and Verification Knowledge

Knowledge class	Rule and data	Amount of rules or data
Diagnostic Knowledge	rule module	2 modules
	rule	40 rules
	primitive	73 primitives
	data definition	114 data
Verification Meta-Knowledge	test-case set	4 set
	error detection knowledge	9 rules
	knowledge frame	2 frames
	assumption generation knowledge	62 knowledge (generation:50, constraint:12)

(b) Verification result

evaluation case	number of assumptions				processing time (second)
	a	b	c	d	
case-1.specialize	84	53	28	3	43
case-2.specialize	50	26	22	2	51
case-3.generalize	49	38	9	2	28
case-4.alternate concept	101	42	58	1	33

a = b + c + d

a..generated assumptions (rules) by assumption generation knowledge
b..conflict assumptions (rules) detected by ATMS
c..conflict assumptions (rules) detected by inference
d..consistent assumptions (rules) verified by ATMS and inference