62－A002

知的情報処理システムに関する調査研究報告書

エキスパートシステム開発ツールの比較評価

昭 和 62 年 9 月

JIPDEC

財団法人 日本情報処理開発協会
ICOT-JIPDEC AI センター

この報告書は，日本自転車振興会から競輪収益の一部である機械工業振興資金の補助を受けて昭和６２年度に実施した「知的情報処理システムの導入・活用に関する調査研究」の成果の一部をまとめたものであります。

はじめに

産業界での人工知能応用の現状は，ユーザレベルで実用化に手が届く知識システムがいくつか出現してきたが，これまでに開発された知識システムの多くはデモンストレーションの段階，せいぜいプロトタイプの段階に留っているとみられる．その理由として，

1）研究開発に着手してまだ間がなく，開発途上にあるものが多いこと，

2）マシン，言語，ツールなどプログラミング環境が整備されていないこと，

3）人工知能研究者や知識技術者の数が不足し，教育体制も確立されていないこと，

4）知識システム構築の基盤技術が未熟であること，

5）知識システム構築の方法論が確立されていないこと，

などを指摘することができる．

現在の技術水準のもとで，知識システムの開発を成功に結びつけるには，問題の注意深い選択と知識技術者の献身的な努力が不可欠である．知識システムの構築に従事している現場の技術者にとっては，ＡＩツールが利用可能であっても，システムを構築していくうえでの方法論がないために，試行錯誤的あるいは場当たり的なシステムづくりを行っている事実を見逃すわけにはいかない．

このように，現在は，知識システムの研究開発において１つの転回点に差しかかっており，次世代知識システムのあり方を技術的および方法論的に見直す格好の時期と思われる．

このような現状認識に基づいて，ＡＩセンタでは，昨年９月に，知的情報処理システム調査研究委員会を設置し，つぎの２つのテーマを調査研究の対象として選定した．

（1）知識システム方法論の確立

知識システム方法論の確立に向けて，つぎの４つの視点からのアプローチをとった．

1）知識システムの基盤技術

まず第１世代知識システムの限界について，問題の所在を明らかにした．ついで次世代知識システムの方法論の基礎を与える基盤技術について，知的な問題解決機能を実現するうえで，高次推論技術の必要性を指摘するとともに，それらの技術的課題を考察した．

2）知識システムの開発の手順

知識システム開発の一般的な手順を“問題の設定”から“性能の評価”に至る９段階にまとめるとともに，開発に際し留意すべき事項として，プロトタイピング，システム開発の条件，従来技術との接続，知識源，システム関与者の役割などを考察した．

3）応用分野の特徴とアプローチ

知識システムの対象を解析型問題と合成型問題に分け，基本的な接近法の違いを論じた上で，解析型問題を解釈問題，診断問題，制御問題に，合成型問題を計画問題と設計問題に，それぞれ分類し，各問題の特徴，基本タスクおよび問題解決機能を明らかにした．

4）知識システムの事例分析

７つの知識システム開発事例を取り上げ，それぞれについて，システムの目的と機能，システム構成と実現環境，システム開発の手順およびタスクと問題解決機能を比較可能な形でまとめ，2）および 3）との対応を明らかした．

（2）システム開発ツールの評価

システム開発ツールの評価に向けて，つぎの４つの視点からのアプローチをとった．

1）ツールの評価項目の体系化

ツール評価の枠組みを，①ツールの概要，②ツールの機能／知識表現，③開発者／利用者インタフェース，④システムインタフェース，⑤ツールの性能，⑥ツールの利用目的，⑦ツールが得意とする分野の特性，の７つに分類し，この下に評価項目を詳細化した．

2）性能評価用テスト問題の提案

ツールの性能評価を行うためには，適切に定義された問題が必要との認識から，いくつかの人工的問題を設定した．第１の型の問題は実行性能の評価に係わる問題である．第２の型の問題は応用適合性という視点からつくられた解釈型，計画型，設計型の問題である．

3）標準問題によるツールの評価

ツールの総合的評価を行うための標準問題として，レンズの骨組み設計問題（キャノン提供）および電気設備のトラブルシューティング問題（新日本製鉄提供）を取り上げ，３つのツール ART，KEE3.0 および Knowledge Craft によるモデリングと評価を行った．

4）知識システム開発方法論とツールの評価方式とのリンク

今年度後半より，知識システム開発方法論ＷＧの成果の一部をツールの評価方式の枠組みに反映させることを計画している．

最後に，本調査研究に当たって，ご協力いただいた委員はじめ関係各位に感謝の意を表する．

昭和62年9月

知的情報処理システム調査研究委員会

委員長	小林重信	東京工業大学・大学院　総合理工学研究科　システム科学専攻助教授
委　員	関根史麿	㈱花王　知識情報科学研究所
委　員	菊地一成	キヤノン㈱　情報システム研究所　知識工学研究部　第２研究室
委　員	湯井勝彦	新日本製鐵㈱　君津製鐵所　設備部　電気計装技術室　室長
委　員（主査）	寺野隆雄	㈶電力中央研究所　経済研究所　情報システム部　知識処理研究部　主査研究員
委　員	山崎知彦	㈱豊田中央研究所　７１研究室　７１２グループ　研究員
委　員	千吉良英毅	㈱日立製作所　システム開発研究所　第２部　１０３ユニット
委　員	森　啓	日本電気㈱　Ｃ＆Ｃシステム研究所　応用システム研究部　主任
委　員	中島俊哉	富士通㈱　教育事業部　教育部　第４教育課
委　員	小林慎一	㈱三菱総合研究所　情報システム部　知識システム室　主任研究員
委　員	福島正俊	三菱電機㈱　中央研究所　システム研究部　第２グループ　グループマネージャー
委　員	森谷正晴	新日本製鐵㈱　君津製鐵所　設備部　電気計装技術室　部長代理
委　員	桃井茂晴	日本電信電話㈱　ＮＴＴ情報通信処理研究所　知能処理理研究室　主任研究員
委　員	畝見達夫	東京工業大学　総合理工学研究科　システム科学専攻助手
オブザーバ	生駒憲治	㈶新世代コンピュータ技術開発機構　研究計画部　調査役
オブザーバ	大野泰治郎	㈱情報数理研究所　ＡＩ開発部
事務局	平井吉光	㈶日本情報処理開発協会　ＡＩ振興センター

システム開発方法論ワーキンググループ

委員（主査）	小林重信	東京工業大学・大学院　総合理工学研究科　システム科学専攻助教授
委員	湯井勝彦	新日本製鐵㈱　君津製鐵所　設備部　電気計装技術室　室長
委員	山崎知彦	㈱豊田中央研究所　７１研究室　７１２グループ　研究員
委員	森　啓	日本電気㈱　Ｃ＆Ｃシステム研究所　応用システム研究部　主任
委員	千吉良英毅	㈱日立製作所　システム開発研究所　第２部　１０３ユニット
委員	中島俊哉	富士通㈱　教育事業部　教育部　第４教育課
委員	小林慎一	㈱三菱総合研究所　情報システム部　知識システム室　主任研究員
委員	福島正俊	三菱電機㈱　中央研究所　システム研究部　第２グループ　グループマネージャー
オブザーバ	藤井祐一	㈶新世代コンピュータ技術開発機構　第５研究室　室長
事務局	平井吉光	㈶日本情報処理開発協会　ＡＩ振興センター

ツール評価ワーキンググループ

委員（主査）	寺野隆雄	㈶電力中央研究所　経済研究所　情報システム部　知識処理研究部　主査研究員
委員	関根史麿	㈱花王　知識情報科学研究所
委員	菊地一成	キヤノン㈱　情報システム研究所　知識工学研究部　第２研究室
委員	森谷正晴	新日本製鐵㈱　君津製鐵所　設備部　電気計装技術室　部長代理
委員	桃井茂晴	日本電信電話㈱　ＮＴＴ情報通信処理研究所　知能処理研究室　主任研究員
委員	畝見達夫	東京工業大学　総合理工学研究科　システム科学専攻助手
オブザーバ	生駒憲治	㈶新世代コンピュータ技術開発機構　研究計画部　調査役
オブザーバ	大野泰治郎	㈱情報数理研究所　ＡＩ開発部
事務局	平井吉光	㈶日本情報処理開発協会　ＡＩ振興センター

知的情報処理システムに関する調査研究報告書

－エキスパートシステム開発ツールの比較評価－

目　次

エキスパートシステム開発ツールの比較評価

まえがき

- Evaluation requires a multi-dimensional framework.
- Danger of comparing apples with oranges just because both are called "tangerines"

by Clancy, W.J., and Rothenberg, J. "Evaluating Expert System Tools." from Tutorial of the 6th AAAI (July.1987).

最近、市販のエキスパートシステム開発ツールの種類は我国で入手できるものにかぎっても、パーソナル・コンピュータ用からメインフレームの大型コンピュータ用にいたるまで、数十種類を越える状況になっており、機能的にも性能的にもさまざまなレベルのものが混在している。その結果、システム開発においてツールのはたす役割が次第に重要になってきていが、このような状況のもとで利用者が適切に自分の問題に合わせてツールを選択することは難しい。本委員会は、ツールを評価する一般的な枠組みを確立することを目的に、昨年度より、調査・検討を続けてきた。

我々は、エキスパートシステム開発ツールの評価そのものは、ツールの提供者、あるいは、利用者の問題であり、単純に解が一つに定まるものではないと考えている。そこで、ツールを評価するにあたって我々のとったアプローチは次の４点にまとめられる（図１）。

①ツールを評価するための項目を、各ツールの概念に依存しない形で、できるだけ詳細なレベルまで記述する。

②個々のツールの性能を測定できる、小規模でwell-definedな性能評価用テスト問題を提案する。

③現実にエキスパートシステムで解かれている問題を簡略化し、比較的大規模な標準問題を、合成型・分析型について２つ設定する。そして、それらを、代表的なツールによって実現することで各ツールの機能・性能を比較検討する。

④エキスパートシステム開発方法論とツールの評価方式とをリンクさせる。

第２部は、これらの内容について中間的に述べるものである。以下に第２部の構成を示す。

１章は、ツールの評価項目と性能評価用テスト問題・標準問題について論じる。まず、ツールの評価項目と性能評価用テスト問題とについて、基本的な考え方を示し、次に、標準問題について、問題の設定にいたった背景について述べる。さらに、１章の付録には、設定した評価項目の一覧表と性能評価用テスト問題の仕様・解説とを掲載する。これらの付録は、本報告を利用する上で読者の役に立つものであると考えている。

２章は、２つの標準問題を３種類の代表的なエキスパートシステム開発ツールで実現した結果を述べる。２つの標準問題は、レンズの骨組み設計問題（合成型の問題）と電気設備のトラブルシューティング問題（分析型の問題）である。これらは、ともに、現実の問題を扱っており、すでに実際にエキスパートシステムを利用して解かれている問題を簡略化したものである。これらの問題は、ＫＥＥ，ＡＲＴ，Knowledge Craft(KC) という代表的な３つのツールを用いてそれぞれインプリメントされ、あわせて６つのプロトタイプシステムが実現された。２章では、まず、２つの標準問題について解説し、その基本タスク・問題解決機能が各ツールでどのように表現されたかを示す。次に、各ツールの開発環境・利用者インタフェースについて述べた後、各ツールのプログラム記述レベル、インプリメンテーションレベルの評価を述べる。２章の付録には、標準問題の仕様とインプリメント結果の概要とを掲載する。

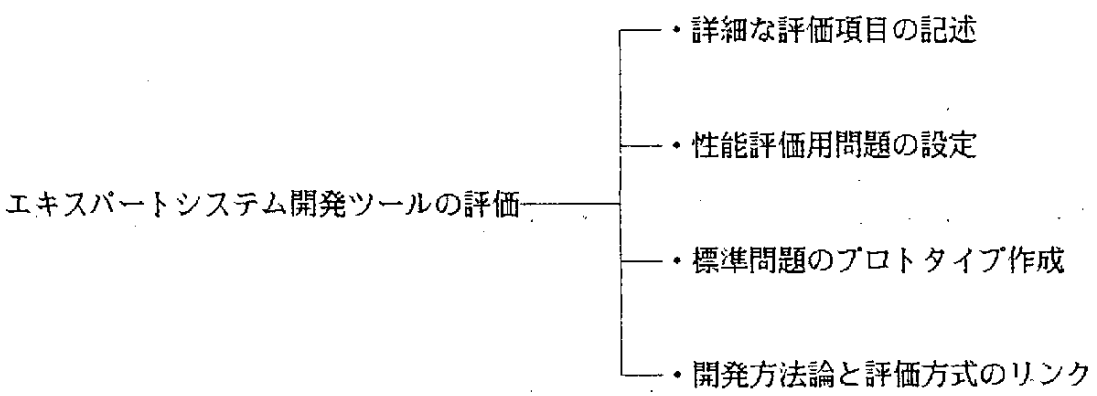


図１．　エキスパートシステム開発ツール評価のアプローチ

1. ツールの評価項目と性能評価用テスト問題・標準問題

1．　ツールの評価項目と性能評価用テスト問題・標準問題

1．1．はじめに

シェルの評価を目的とした調査研究はこれまでにもいくつかなされている。たとえば、[Harmon 87],[日経 85], [Gilmore 86],[Richer 86], [Wall 85]では、知識処理の手法と代表的なシェルの機能とを解説し、各ツールの特性を簡単な表にまとめて示している。これらはシェルのカタログから得られる情報が中心である。複数のシェルを用いて、実際にプロトタイプを開発した上で、これらの評価を実施した報告として[Beach 87],[NASA 87], [Faught 85] がある。[Beach 87]は、共通な小規模な標準問題を設定してその解をリストつきで掲載していること、[NASA 87] は、より大規模なシステムの開発経験を報告するとともに、シェルの実行性能データを公表していること、ならびに、[Faught 85] はシェルの機能拡張という立場から考察を行っていることにそれぞれの特徴がある。我々のアプローチは、これらを統合した、ツール評価の標準的な枠組みを確立しようというものである。これらの評価方式は図１．１のようにまとめられる。

１章では、エキスパートシステム開発ツールについて、評価項目の基本的な考え方とツール評価項目の一覧とを示し、次に、実際の性能評価のために設定した人工的なテスト問題について述べ、各性能評価用問題の仕様と解説を示す。さらに、現実の問題に即してとりまとめた比較的大規模な標準問題についてその評価の視点と実作業における位置づけについて論ずる。最後に、提案した、評価項目、問題の利用方法について考察する。

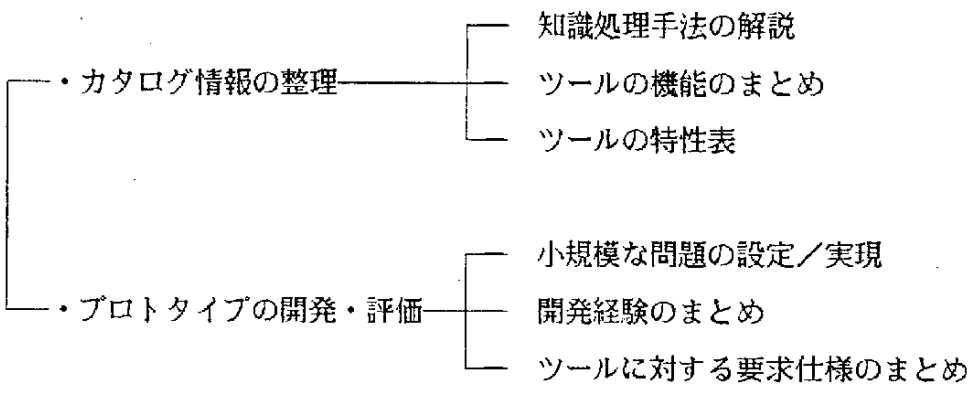


図１．１．　従来のツール評価方式のまとめ

参 考 文 献

[Harmon 85]]Harmon,P.,and King,D.: "Expert Systems -ArtificialIntelligence in Business." John-Wiley,1985.

[日経 85] 日経エレクトロニクス：" 商品化相次ぐエキスパート・システム開発用ツールを比較する." 1985.11.4.pp.153-175.

[Gilmore 86] Gilmore,J.F., Pulaski,K., Howard,C.: A Comprehensive Evaluation of Expert System Tools. Proc.of SPIE,Vol.635 (Applications of Artificial Intelligence III), pp.2-16(April 1986).

[Richer 86] Richer, M.H.: An Evaluationof Expert System Development Tools. Expert Systems, Vol.3, No. 3, pp. 166-183 (July 1986).

[Wall 85] Wall, R. S., et al.: An Evaluation of Commercial Expert System Building Tools. Data & Knowledge Engineering, Vol. 1, No. 4 pp. 279-304 (1985).

[Beach 87] Beach, S.S.: "Analysis of High-End Expert Systems Tools." Spang Robinson Report, 1987.

[NASA 86] NASA: "Expert Systems Handbook." Johnson SpaceCenter Report, 86-FM-19, JSC-22230,July 1986.

[Faught 85] Faught,W.S.: "Functional Specifications for AI Software Tools for Electric Power Applications." EPRI Report NP-4141, Research Project 2582-1, August 1985.

寺野 隆雄（（財）電力中央研究所）

１．２．エキスパートシステム開発ツールの評価項目

１．２．１．概　要

現在発表されているツールの多くは、ルールやフレームの概念に基づいて構成されているが、現実には各概念の実現方法が個々のツールによって微妙に異なっている。そのため、ツールを適切に評価するためには、特定のツールに特化された概念をきめ細かく調査することが必要となる。また、ツールを利用するという立場からは、応用分野の特性に関する考察も必要となる。

そこで、我々は、ツールの評価項目の枠組みを、①ツールの概要、②ツールの機能／知識表現、③開発者／利用者インタフェース、④システムインタフェース、⑤ツールの性能、⑥ツールの利用目的、⑦ツールが得意とする分野の特性、の７つに分類して定めることとした（図１．２－１）。そして、各評価項目は、重複をいとわずにできるだけ詳細なものとし、また、個々のツールで用いられる特有の概念に左右されないように定めた。さらに、現在のツールでは提供されていない機能・項目についても、人工知能研究の立場から今後重要となると考えられるものについては積極的に記述することとした。これらの評価項目は、文献[Harmon 87],[日経 85], [Gilmore 86],[Richer 86], [Wall 85]で示されている項目を完全に含んでいる。したがって、現在入手できるツールで標準的に採用されているルールやフレームなどの項目については非常に詳細なレベルまでの記述が可能となっている。一方、そのような従来型のツールの概念では捉らえるのが難しいような機能をもつツールについては、問題解決レベルや基本タスクの項目から必要な評価が得られるようにした。さらに、ツールと、[Chandrasekaran 86],[小林 86] などに見られる開発方法や基本タスクとの関連性にも留意している。

具体的な項目は、次節に記述したとおり膨大なものであるが、これによって、ツールのベンダーにとってもユーザにとっても、共通の評価の枠組みを設定できるものと考えている。ただし、これらの評価項目は、どのようなツールに対しても適用可能なことをねらって設定したものであるので、ツールの実現技術上、性能が互い矛盾する項目も含まれている。たとえば、ツールの機能の豊富さと実行速度との間には必ずある程度のトレードオフが存在する。したがって、必ずしも、すべての項目について○印がつくことがよいツールの条件を表わすわけでない。むしろ、ツールの利用目的が明確であれば、それに関連す

る項目についてのみ良い評価がなされ、不要な機能が少ないツールの方が望ましいことになる。

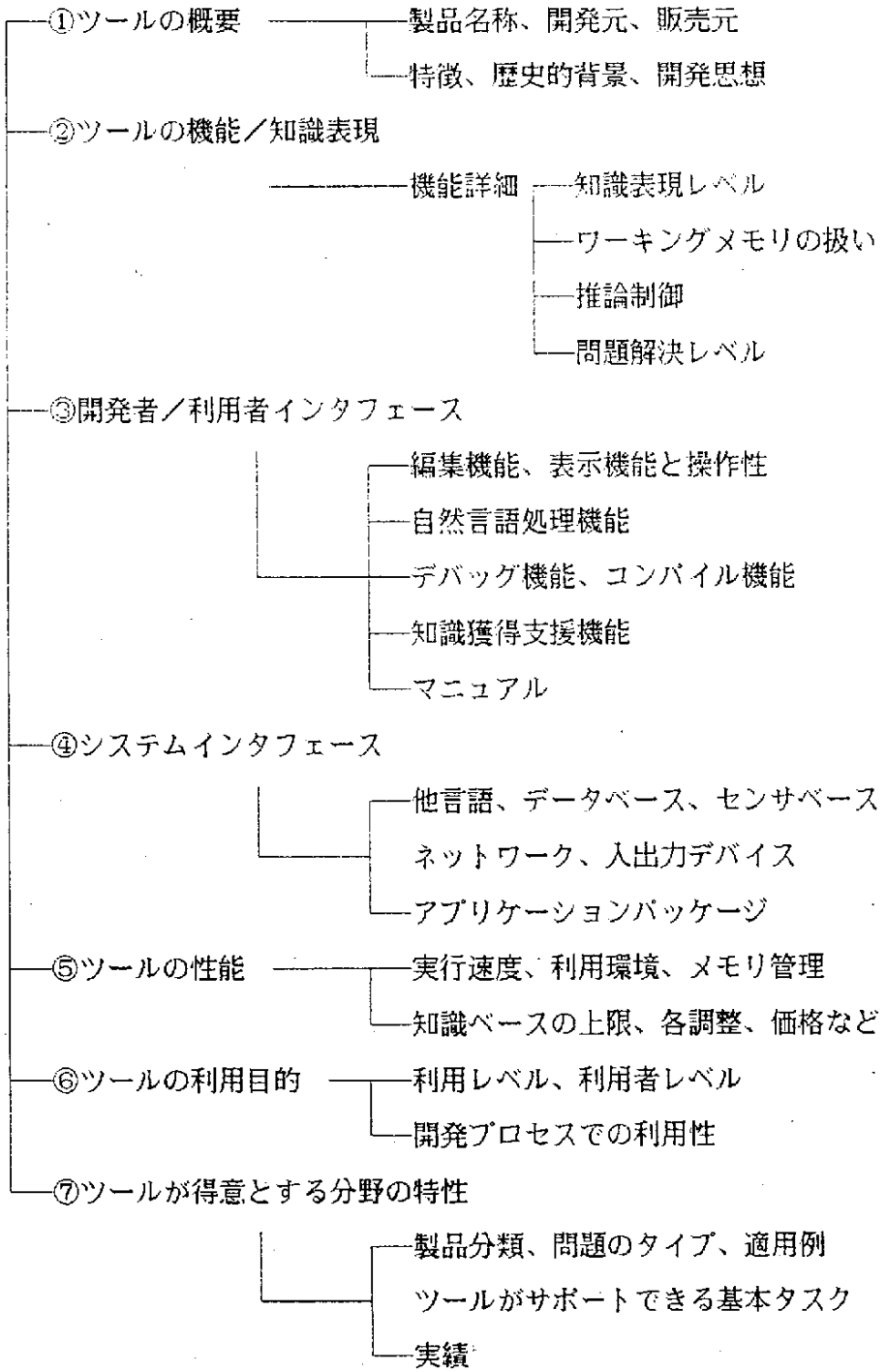


図１．２－１．ツール評価項目の分類

参 考 文 献

[Harmon 85]]Harmon,P.,and King,D.: "Expert Systems -ArtificialIntelligence in Business." John-Wiley,1985.

[日経 85] 日経エレクトロニクス：" 商品化相次ぐエキスパート・システム開発用ツールを比較する." 1985.11.4,pp.153-175.

[Gilmore 86] Gilmore,J.F., Pulaski,K., Howard,C.: A Comprehensive Evaluation of Expert System Tools. Proc.of SPIE,Vol.635 (Applications of Artificial Intelligence III), pp.2-16(April 1986).

[Richer 86] Richer, M.H.: An Evaluationof Expert System Development Tools. Expert Systems, Vol.3. No. 3, pp. 166-183 (July 1986).

[Wall 85] Wall, R. S., et al.: An Evaluation of Commercial Expert System Building Tools. Data & Knowledge Engineering, Vol. 1, No. 4 pp. 279-304 (1985).

[Chandrasekaran 1986] Chandrasekaran, B.: Generic Tasks in Knowledge-Based Reasoning: High-Level Building Blocks for Expert System Design. IEEE Expert, Vol.1,No.3,pp.23-30 (Fall 1986).

[小林 86] 小林重信：" 知識工学. " 昭晃堂, １９８６.

寺野　隆雄（（財）電力中央研究所）

１．２．２　エキスパートシステム開発ツールの評価項目一覧

以下は、これまでの検討に基づいて設定したエキスパートシステム開発ツールの評価項目一覧である。特別の記述がないかぎり、各項目はｘ，△，○で答えることを前提に設定している。各項目についての詳しい解説については別途報告する予定である。

１．　概　要

１－１）製品名称（略称） １－２）開発元　（販売元）	
１－３）特徴（簡潔に記述する）	
１－５）シェルの歴史的背景と開発思想（簡潔に記述する）	

２．ツールの機能／知識表現

２－１）概　要　　（ｘ，△，○で答える）

－機能の豊富さ －機能間のバランス －記述の統一性 －表現の理解容易性	

２－２）機能の詳細項目

基本的に○をつける。その他の項目があるときは簡潔に記述する。

２－２－１）知識表現レベル

①ルール

－推論方向

前向き

後ろ向き

両方向混在

－ルールの表現要素

・条件部

結合子

and (Implicit/Explicit), or, not

その他の結合子　（　　）

述語記述

論理述語のユーザによる拡張性

手続き

算術演算式

その他の手続き

その他の記述　（　　）

・実行部

手続き

言語仕様としての手続き記述（condition, loop, while, その他）

他言語の利用可能性

その他の記述　（　　）

・パタン照合の対象データ型と方法

非構造データ（ファクト）

データ型　（整数、実数、シンボル、ストリング、
リスト／シーケンス）

方法　（変数、リテラルパタン、ワイルドカード）

構造化データ（ベクター、フレーム、オブジェクトなど）

構造に対して

構造（関係）の指示、参照

関係の制約

フレームの親子関係による制約

構造名

変数として使用可能

指定複数フレーム内の照合

属性に対して

属性名の指定

属性値の指定

－その他の特徴

・デモンの記述

ルールのbefore/after

・実行部の実行制御　（実行保留／解除）

－ルールの制御表現

・ルールのグループ化

各グループの呼出し方法　（　　）

グループのイベント起動機能（ブラックボードの利用、その他）

・ルールの実行制御

②フレームとオブジェクト

－関係の記述

・継承関係

継承関係の有無

継承関係の種類

class-subclass

class-instance

その他の関係（　　）

多重継承の有無

・継承の制御方法（概要：　　）

継承単位の指定（フレーム単位に指定可

スロット単位に指定可）

制御方法

（デフォルトとして）親の値

（必ず）親と同値

その他

多重継承の制御（　　）

・関係のユーザ定義の可能性

継承関係の拡張性

非継承関係の拡張性

－属性／データの記述

・属性値の型

シングルバリュー（整数、実数、シンボル、ストリング、その他）

マルチバリュー　（リスト／シーケンス、その他）

その他（　　）

・スロットの制御（ファセットなど）

属性値の型の制限

属性値の値の制限

属性値の数の制限

スロットの存在するフレーム／オブジェクトの範囲の制限

その他

・情報隠蔽機能

－手続きの記述

・付加手続きの有無

種類（if-needed(get-slot), if-added(put-slot),
if-removed, その他）

・メソッドの有無

メソッドの記述言語（　　）

ルールの記述

メソッドの継承の有無

多重継承の有無

多重継承の制御（　　）

・起動操作（メッセージパッシング、デーモン、その他）

メッセージパッシングの方法・形式（　　）

実行の並列性

③論理表現の有無

－言語の種類（Ｐｒｏｌｏｇ、その他）

・バックトラック機能

・ユニフィケーション機能

④他言語とのインタフェース

－言語名：（　　）

－方式

⑤不確実な情報の扱いの有無

－確信度（ｃｆ）

計算方法（mycin型、その他）

ユーザ定義可能性

－ファジィ

メンバーシップ関数の種類

演算の種類

ユーザ定義可能性

－Dempster/Shafer理論

－その他

２．２．２）ワーキングメモリの扱い

①ワーキングメモリ上のデータ表現
　ファクト，ブラックボード，ベクタ，フレーム、その他

②ワーキングメモリの管理

　ＡＴＭＳ，　ＴＭＳ，　ブラックボード

③多重世界の表現
　コンテクスト，　ビューポイント，　その他

２．２．３）推論制御

①ルール

－競合解消
　決定要素
　　（ルールの優先度、ルールのあいまいさ、データの新しさ、
　　ルール条件部の詳細さ、ルールの記述順序、その他）
　決定要素の適用順序
　適用順序の変更可能性

－推論制御の最適化手法
　競合ルール集合抽出のアルゴリズム
　　（rete，改良rete，その他）

－メタ制御

－メタルール

②フレーム／オブジェクト

－継承解決

－デモン制御

－メッセージパッシングの制御

③アジェンダ機能

2．2．4）問題解決レベル

①探索手法の実現性

－blind search（深さ優先、幅優先）

－heuristic search（最適探索、最良優先、山登り、その他）

－その他（And/Or graph, Game Tree search, その他）

②問題解決のモデル

－推論モデル（不確実な推論、仮説的な推論、時制推論、分散協調推論、定性推論、モデルベース推論、その他）

－データの扱い（非同期入力、リアルタイム性）

③知識処理手法の実現性

(generate-and-test, hierarchical-generate-and-test, guessing, topdown-refinement, least-commitment, constraint-propagation, backtracking, dependency-directed-backtracking, opportunistic-scheduling, classification, その他)

3．開発者／利用者インタフェース

3．1） 概要（x，△，○で答える）

	システム開発者	システム利用者
・操作性 ・統合性 ・一貫性 ・拡張性 ・説明機能 ・ヘルプ機能（オンライン？） ・エラー処理機能		

３．２）編集機能

	編集	ブラウズ
・編集対象の種類 テキスト、図形、その他		
・専用エディタ／ブラウザの有無		
・シンタックスチェックなどの一貫性管理		
・知識ベースの世代管理		
・拡張性		
・操作性，スピード		
・説明機能		
・一括編集機能		－
・コンパイルチェック		－
・エラー処理		－
・デバッグ機能		－
・複数人による開発サポート		－
・プロジェクト管理		－

３．３）表示機能と操作性

	システム開発時	システム利用時
・ウィンドウシステム －グラフィック －アイコン		

－ポップアップメニュー －ズーミング －アニメーション		
・カラー表示		
・シミュレーション機能		
・グラフィック（２次元／３次元）		
・知識ベースのグラフィック表示／編集 （ルール、フレーム、その他）		

３．４）自然言語処理機能

	システム開発時	システム利用時
①対象言語 －日本語（入力／出力） －英語（入力／出力） －その他（入力／出力）		
②表示機能 －テキスト －アイコン／メニュー		

③自然言語理解 －字句解析 －構文解析 －意味解析 －文脈解析		
④音声処理、会話理解		

3．5）デバッグ機能

・専用デバッガの有無
・知識ベースの一貫性管理
・拡張性
・操作性，スピード
・説明機能

－アジェンダ
－マッチング
－推論過程のトレース
－ジャスティフィケーション
－ＷＨＹ
－ＨＯＷ
－ｗｈａｔ－ｉｆ
－その他

・エラー処理

・テスト機能

－実行過程のセーブ機能
－テストデータの保存、管理、再実行
－テストケースの自動生成

・チューニング機能

－パフォーマンス測定
－ボトルネックの発見のサポート
－知識ベースの変更サポート

3－6）コンパイル機能

－インタプリタ／コンパイラ両モードの併用
－インクリメンタルコンパイル
－シンタックスチェック
－基本的なセマンティックチェック

3－7）知識獲得支援機能

－開発方法論のサポート
－知識表現（ルールなど）の自動生成
－モデリング支援機能
－学習機能

3－8）マニュアル

－入門マニュアル
－リファレンスマニュアル
－その他マニュアル
－ツールに即した教科書の有無
－ツールに即した教科書の名前
－例題（ディスク、テキスト、その他）

4．システムインタフェース

4－1）他言語インタフェース

- 有無
- 種類
- 入出力パラメタの受け渡し
- デモンの記述

4－2）データベースインタフェース

- 有無
- 種類
- 方法

4－3）センサベースインタフェース

- 有無
- 種類
- 方法
- オンライン機能
- 非同期処理機能

4－4）ネットワークインタフェース

- 有無
- 種類
- 方法

4－5）~~入出力デバイスインタフェース~~

- 有無
- 種類
- 方法（タッチパネル、ジョイスティックなど）

4－6）アプリケーションパッケージ・インタフェース

- 有無
- 種類
- 方法

4－7）その他

5．ツールの性能

5－1）実行速度

対象ハードウェア：

	ルール	フレーム	BB／ATMS	その他
インタプリタ				
コンパイラ				
（単位：logical instruction/sec） （付記：性能評価条件：　　）				

5－2）利用環境

	名称	OS	開発言語	最小容量	
				ディスク	メモリ
lisp machine workstation personal computer main frame その他					

5－3）　メモリ管理：　OS／独自

ガーベッジコレクションの方式：

5－4）　（入門／教育用の場合）

知識ベースの上限：

ルール

フレーム

その他

合計

5－5）　拡張性

・ソースプログラムの提供

・推論機構の変更

・知識表現の拡張性

5－6）　価格等

・価格　　1set:

2set以上：

・保守費用

（これらの項目は複数のハードウェアで稼働する場合は
ハードウェアごとに記述する）

・トレーニング費用　　入門

高度

その他

・コンサルタント費用／方法

（複数あればそれぞれについて記述する）

6．ツールの利用目的

6－1）　利用レベル

ＡＩ技術の勉強／教育 フィージビリティスタディ プロトタイプ開発 実用システム開発

6－2）　利用者レベル

・記号処理言語の経験の必要性　Lisp Prolog その他	
・使用者レベル （○をつける）	エンドユーザ 知識技術者 人工知能研究者 その他
・開発の委託の可能性／容易さ	
・トレーニング：	ＯＪＴ 教育プログラムの有無・特徴
・使用時のサポート：	Ｏｎ－Ｃａｌｌ その他
・Version-up，バグ情報の種類、頻度	
・ユーザ会の有無、頻度、内容	

6－3）開発プロセスでの利用性（どのレベルでツールが使用できるか記述する）

問題の設定、既存技術の評価、知識源の同定、専門家モデルの同定、
ユーザモデルの同定、知識表現の選択、知識の移植、知識ベース管理、
性能評価、システムの拡張

7．ツールが得意とする分野

7－1）製品分類

汎用ツール
専用ツール／アプリケーションパッケージ
（適用分野　　）

問題のタイプ

合成型問題：　計画型　（適用例　　）
　　設計型　（適用例　　）

解析型問題：　解釈型　（適用例　　）
　　診断型　（適用例　　）
　　制御型　（適用例　　）

その他

7－2）ツールがサポートできる基本タスク（○をつける）：

合成型問題：	計画型	計画過程の階層化、組合せ的探索、制約条件の相互作用、環境の予測、知的バックトラッキング、計画事例の検索／利用、計画の評価、（その他　　）
	設計型	構成要素とその関連の表現、設計問題の階層的表現、代替案の生成、部分システムの評価、知的バックトラッキング、設計事例の検索／利用、並列的問題解決、（その他　　）
解析型問題：	解釈型	特徴抽出、モデルとの照合、システム構造の同定、システム状態の推定、あいまいさの処理、不完全性の処理、解釈の多様性の処理、（その他　　）
	診断型	事象の分類／階層的表現、システム構造の階層的表現、計測点の選択／決定、異常原因の同定、浅いモデル、深いモデル、（その他　　）
	制御型	システム構造の表現、システム動特性の表現、状態の予測、安定化操作、低コスト操作、ガイダンス、（その他　　）

7－3）実績

- 販売実績　　－　国内、全体、合計
- 実用システム　－　国内、全体、合計
- 代表的なシステムとその特徴
- 主要ユーザ（国内、国外）

１．３．性能評価用のテスト問題の提案

１．３．１　概　要

ツールの性能を評価するためには、そのツールで提供される人工知能の概念の特性に即した、人工的、well-definedかつ小規模な問題が必要となる。現在、よく用いられる問題には、Monkey-Banana Problem、農夫のジレンマなどがあるが、これらは、小規模すぎる、問題のサイズをパラメタで設定できない、などの欠点がある。そこで、我々は、推論機構の実行性能をルールシステム、フレームシステムについて評価できる単純な問題を設定した。さらに、エキスパートシステム化の要請が強い、解釈型、設計型、計画型の各問題について、その特性をよく表現する３つの問題を作成した。各問題の記述は、付録２にまとめて示したとおりであるが、これらは、名称、目的、問題の定義・解説、留意事項、例の項目からなる。

ルールシステムに関する実行性能評価問題は、ルール・条件の個数とルールの条件照合に要する時間との関係を測定することを目的としており、Reteアルゴリズムをコンパイラに採用しているツールの性能を知る場合に有用である。フレームシステムに関する実行性能評価問題は、システムに存在するフレームの個数とそのアクセス時間との関係を継承なし／継承ありで測定できるように設定した。ただし、フレームシステムについては、Reteのような決定的なアルゴリズムが現在のところ存在しないので、設定した問題はツールのインプリメントに採用たアルゴリズムの差を検出できるほど精密なものではない。

解釈型、設計型、計画型の各問題は、それぞれ人手で解くことができる程度のごく小規模なものであるが、いずれも、具体的な知識表現方法やルールは提示していないので、インプリメンタの能力とツールの性能とによってかなり異なる結果が得られるものと思われる。ただし、これらの問題は、いずれも条件を精密にしていけば実用上の問題にも適用できる概念を含んでおり、その意味では、テスト問題として適切なものといえよう。解釈型問題は特にパターンマッチングの知識をどう実現するかが課題となる。計画型問題は実行可能解をひとつ生成することを目的としているが、これに種々の制約を加えることで実用的なものとなり、また、解の評価を行う最適型の問題にも拡張することができる。設計型問題は宣言的な知識の記述方法が課題となる。

これらの問題は、ツールの性能を比較評価する場合に有用であると同時に、知識処理

技術やエキスパートシステムの教育時にも利用できるものである。

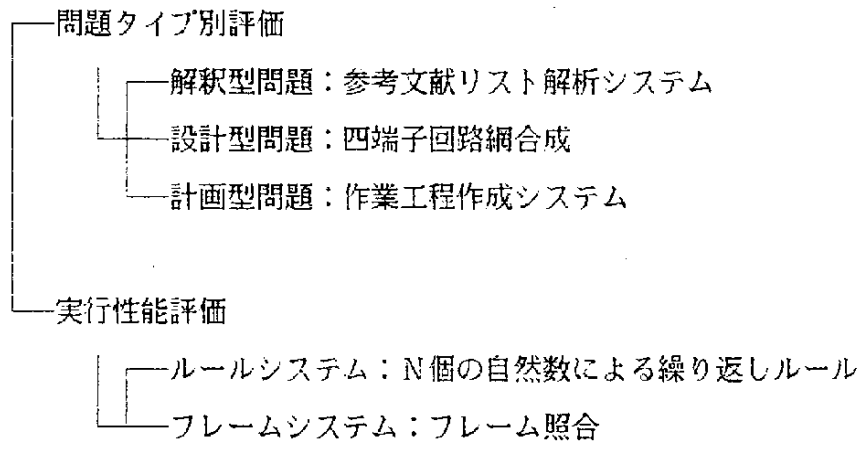


図１．３－１．　性能評価用テスト問題の分類

寺野　隆雄（（財）電力中央研究所）

１．３．２　解釈型問題

名称

参考文献リスト解析システム

目的

文字列領域のデータについて、それに含まれる部分文字列の分類を行ない、全体の構造を抽出する。個々の分類カテゴリーに対応する特徴記述能力、および、それらの統合、競合解消機構の記述能力を試す。

問題の定義

論文の末尾に付されている参考文献リストを入力とし、文献データベースの内容のような、表題、著者名など項目別に整理した構造を出力する。

問題を単純化するために、処理の対象となるデータをつぎのように制限する。

①１度に１つの文献を表わすデータのみを処理する。

②単行本の１つ章を指すようなデータは対象外とする。

システムの仕様、および実現上の技術的要点は次のとおり。

(1) 与えられた文字列データに含まれる部分文字列を既知のカテゴリーに分類する。

カテゴリーとして、①著者名、②論文題目、③出版社名、④雑誌名、⑤学位名、⑥会議名、⑦報告書名、⑧機関名、⑨部門名、⑩巻、⑪号、⑫頁、⑬発行年　を考える。その他は無視してもよい。

(2) １つのデータは、おおむね次の図1.3-1 に示す構文に従うと考えてよい。

ただし、発行年は、著者名の後や巻の後などに来ることもある。また、途中に地名や発行月など上記以外の項目が挿入されている場合もある。

(3) 各カテゴリー毎に、データの特徴として、キーワード、少数種類の単語、文法などが使え、各カテゴリーに適した特徴把握手続きをコード化する必要がある。

各カテゴリーの特徴はおおよそ次のようなものである。

①構文によるもの：著者名、論文題目、巻、号、頁、発行年

例えば、著者名の構文は図1.3-2 のようなものと考えてよい。ただし、この構文は、よく使用される著者名の記述法を表現するに過ぎないという点に注意する必要があ

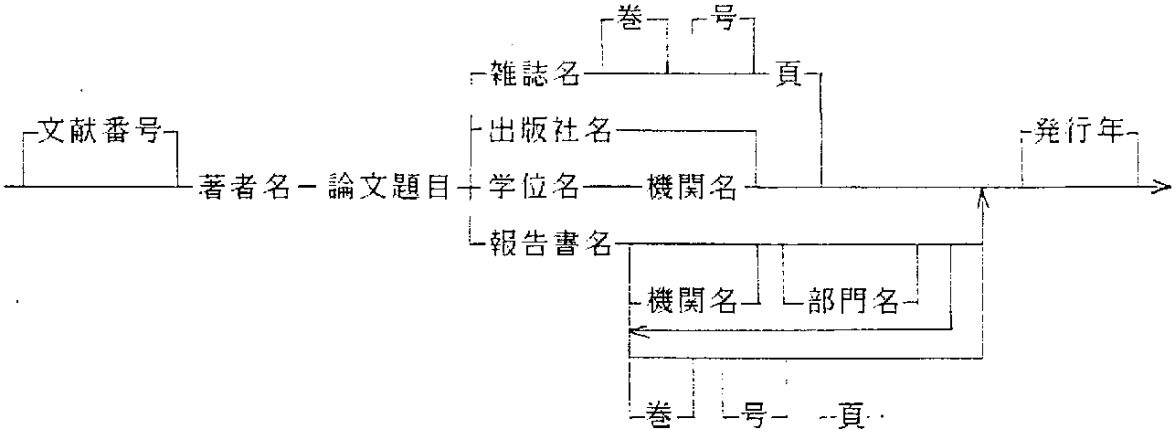


図1.3-1 1つのデータの構文

著者名：

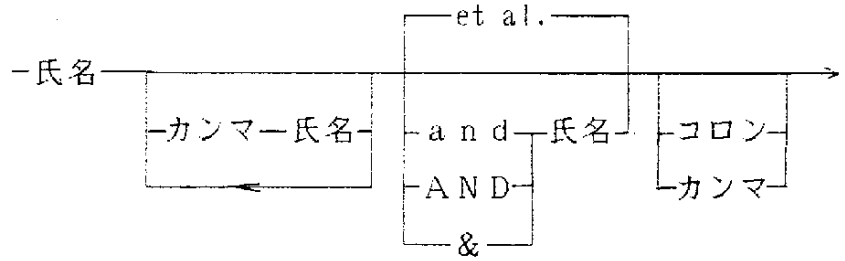


図1.3-2 著者名の構文の例

る。この構文に従う文字列が必ず著者名になる、あるいは、この構文に従わないものは著者名ではないとは限らない。

②キーワードによるもの：学位名、会議名、報告書名、所属機関名、所属部門名

学位名：Thesis, Ph.D., Mc.D., Sc.D., …

会議名：Preceedings of, Proc., Conf., Progress, …

報告書名：Report, Rept., TR, …

機関名：University, Univ., School, Institute, Laboratory, Lab., …

部門名：Department, Dept., …

③辞書との照合によるもの：出版社名、雑誌名

データ中に出現する出版社名、雑誌名を辞書に登録しておき、照合する。

(4) 互いに類似の特徴を持つカテゴリー（例：地名と人名、頁と年号など）も存在するが、個々の特徴についての確からしさや全体の文脈で判断がつく場合が多い。

留意事項

解析の対象となる参考文献リストの形式は雑誌などによって様々なものがあるが、できる限り多くの種類の形式を解析可能とするほうが望ましい。

データの性質から考えて、自然言語の構文解析に用いられるような、探索に基づいて構文木を組み上げるといった手法よりは、個々の項目毎に、その特徴を満足する部分文字列を切り出すモジュールを起動し、相互の調整を取るような分散協調型の機構を採用する方が好ましい。

実現例としてＭＡＲＣＳ［Kobayashi et al. 84］を参考にするとよい。

例

以下は入出力の対の例である。

入力： [1] M.BLUM,R.W.FLOYD,V.R.PRATT,R.L.RIVEST AND R.E.TARJAN,"Time bounds for selection," J.Comp.Sys.Sci.,7(1972),pp.448-461.

⇒

出力： AUTHOR : M.BLUM R.W.FLOYD V.R.PRATT R.L.RIVEST R.E.TARJAN
TITLE : Time bounds for selection
YEAR : 1972
JOURNAL : Comp.Sys.Sci.
VOLUME : 7
PAGE : 448-461

入力： [1] Anderson,R.M. (1981) Core Theory with Strongly Convex Preferences, Econometrica 49 1457-1468.

⇒

出力： AUTHOR : R.M.Anderson
TITLE : Core Theory with Strongly Convex Preferences
YEAR : 1981
JOURNAL : Econometrica
VOLUME : 49
PAGE : 1457-1468

入力： [1] D.Ohne,Topics in nonlinear filtering theory,Sc.D.Thesis,
Massachusetts Institute of Technology (1980).

⇒

出力： AUTHOR : D.Ohne
TITLE : Topics in nonlinear filtering theory
YEAR : 1980
THESIS : Sc.D.Thesis
ORGANIZATION : Massachusetts Institute of Technology

入力： [1] P.S.Kirshnaprasad and S.I.Marcus,System identification
and nonlinear filtering:Lie algebras. Proc.IEEE 20th
Conf. on Decision and Control,San Diego,CA,pp.330-334 (1981).

⇒

出力： AUTHOR : P.S.Kirshnaprasad S.I.Marcus
TITLE : System identification and nonlinear filtering:Lie algebras
YEAR : 1981
CONFERENCE : IEEE 20th Conf. on Decision and Control
PAGE : 330-334

入力： [1] J.A.Appels,et al.,"Local oxidation of silicon and its
application in semiconductor device technology,"Phillips
Research Reports,Vol.25,no.2,1970,pp.118-132.

⇒

出力： AUTHOR : J.A.Appels,et al.
TITLE : Local oxidation of silicon and its application in semiconductor
device technology
YEAR : 1970
REPORT : Phillips Research Reports
VOLUME : 25

NUMBER : 2

PAGE : 118-132

入力： [1] A.V.Fiacco and G.P.McCormick. Nonlinear Programing: Sequential Unconstrained Minimization Techniques. John Wiley and Sons (1968)

⇒

出力： AUTHOR : A.V.Fiacco G.P.McCormick

TITLE : Nonlinear Programing: Sequential Unconstrained Minimization Techniques

YEAR : 1968

PUBLISHER : John Wiley and Sons

（担当　畝見 達夫）

参考文献

[Kobayashi et al. 84] 小林重信，畝見達夫，望月浩史，参考文献解析エキスパートシステム，情報処理学会，知識工学と人工知能研究会資料３４－６（1984）.

１．３．３　設計型問題

名称

四端子回路網合成

目的

要求仕様として与えたシステムの特性から、これ満足するシステムの構成要素の組み合わせ構造とその各構成要素の仕様の決定を行う簡単な設計型の問題において、ツールの機能・性能等の比較評価をする。

問題の定義

（１）ゴール規則

要求仕様は、以下のシステムの特性を示す行列Ｆｓとして与えられる。

$$Fs = \begin{vmatrix} As & Bs \\ Cs & Ds \end{vmatrix}$$

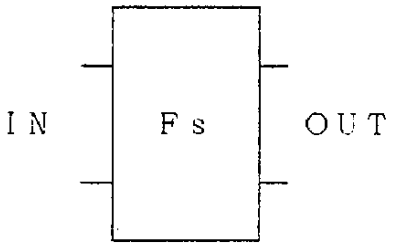


図１．システム

この行列Ｆｓを満足する、システムの構成とその構成要素を探索する。

ただし、

①行列Ｆｓの各要素Ａｓ，Ｂｓ，Ｃｓ，Ｄｓの値は、ユーザ入力で与えられる実数とする。

②要求仕様として与えられた行列Ｆｓ内の各要素Ａｓ，Ｂｓ，Ｃｓ，Ｄｓの値と設計されたシステムのそれらの間には、それぞれ５％以内の範囲で誤差を許容する。

③求める構成は、構成要素数の最も少ないものとし、同一構成要素数のものが複数存在する場合は全て求め出力する。

④結果の出力項目は、後述のシステムの各構成要素の仕様の値2とその構成法（直列・並列の区別）および構成順である。

（2）構成要素の使用規則

使用出来る構成の仕様は、パラメータｚのみで定義される（図2）。このパラメータの値は、以下のいずれかの値をとる必要がある。

Ｚ＝ｋ＊１０^{n}

ここで、ｋ＝2.7，3.3，3.9，4.7，5.6，6.8，8.2，10，12，15，18，22

および、ｎ＝0，1，2，3，4，5，6

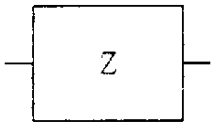
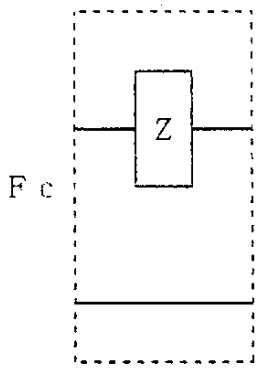


図2．構成要素

（3）構成要素による部分システムの生成規則

１つの構成要素から、部分システムを構成出来る。この部分システムは、構成法によって、直列構成と並列構成に分けることが出来る（図３－１、図３－２）。



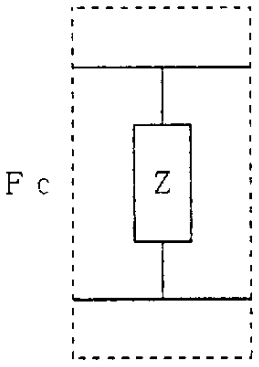


図３－１．直列構成による部分システム　図３－２．並列構成による部分システム

それぞれの構成法から出来る部分システムの特性を示す行列Ｆｃは、構成要素の仕様パラメータｚとの間に以下の関係がある。

$$Fc = \begin{vmatrix} Ac & Bc \\ Cc & Dc \end{vmatrix}$$

のとき

①直列構成の場合、Ａｃ＝１、Ｂｃ＝Ｚ、Ｃｃ＝０、Ｄｃ＝１

②並列構成の場合、Ａｃ＝１、Ｂｃ＝０、Ｃｃ＝１／Ｚ、Ｄｃ＝１

（4）部分システム間の合成規則

1列に構成された部分システムにより、システムは構成される。設計されたシステムの特性を示す行列は、部分システムの構成順に全ての部分システムの特性を示す行列の積を行ったものとなる。

したがって、もし部分システムuとv隣接して構成すれば、両者を合成した合成部分システムw（もしくはシステム）を構成できる（図4）。

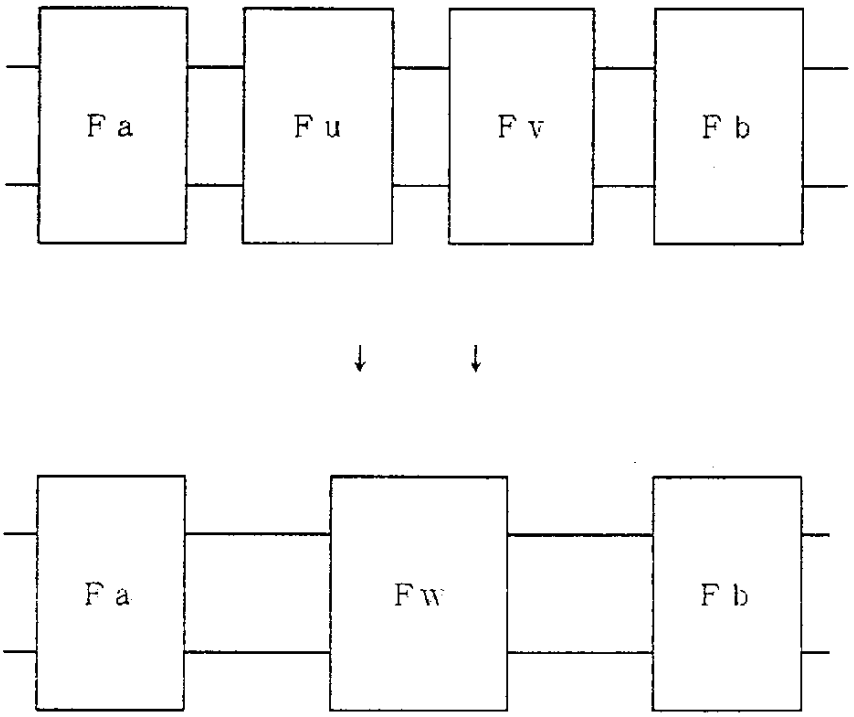


図4．部分システム間の合成

このとき合成部分システムwの特性を示す行列Fwは、部分システムuとvの特性を示す行列FuとFvの間に以下の関係がある。

$$Fw = Fu * Fv$$

$$Fw = \begin{vmatrix} Aw & Bw \\ Cw & Dw \end{vmatrix} = \begin{vmatrix} Au*Av+Bu*Cv & Au*Bv+Bu*Dv \\ Cu*Av+Du*Cv & Cu*Bv+Du*Dv \end{vmatrix}$$

留意事項

（１）出来る限り設計知識は、宣言的かつ明示的に記述されなければならない。

例

（１）要求仕様の入力数値例

実行に先立ち、要求仕様としてシステムの特性を示す行列Ｆｓ内の各パラメータの入力を要求する。このとき、代表値として以下の各パラメータ値の入力する。

Ａｓ＝１．３

Ｂｓ＝２７００

Ｃｓ＝０．００１

Ｄｓ＝２．８

（２）出力の例

求められたシステムの構成により、以下の表示例を行う。

構成の解　１

構成要素　１　入力端からの構成要素順　１　構成法　直列　仕様　Ｚの値

構成要素　２　入力端からの構成要素順　２　構成法　並列　仕様　Ｚの値

構成の解　２

構成要素　１　入力端からの構成要素順　１　構成法　並列　仕様　Ｚの値

構成要素　２　入力端からの構成要素順　２　構成法　直列　仕様　Ｚの値

１．３．４　　計画型問題

名称　　作業工程作成ｼｽﾃﾑ

目的　　簡単な計画問題であるが条件を付け加える事によって十分実用的であり、解の探索問題としても多様な手法を考えさせる。

問題の定義

異なった五つの作業A,B,C,D,Eを異なった４地点P1,P2,P3,P4で同時に進めなければなりません、ﾕｰｻﾞが与えた次の３条件を満たす各地点ごとの１日の作業工程表を作成するｼｽﾃﾑを作ってください。

条件１・・各地点ごとに、作業可能時間が設定されており、各作業はその範囲内でする、

条件２・・各作業にはそれぞれ１日に作業しなければならない時間数が指定されている

条件３・・異なった場所でどうじに並行して作業が出来るかどうかを表わす作業ﾃｰﾌﾞﾙ表があり、その表でXが与えられている組み合わせは同じ時刻に作業が出来ない。

留意事項

(1) 各作業はできるだけ連続して進められるほうが望ましい。

(2) 与えられた条件で作業工程表が作成出来ない場合はそのようにﾒｯｾｰｼﾞを与えること。

(3) 現実の作業を想定し、望ましい解の条件を記述しそれを探索するﾋｭｰﾘｽﾃｨｯｸな手法をあたえよ。（例えば、全体の作業時間を最小にするなど）

例(入力ﾃﾞｰﾀ)

(1)　各場所での作業可能時間帯

```
        0       8       12 13    18       24
P1              --------  ------
P2                   --------  -----
P3              --------  ------
P4                           ---------
```

(2)　各作業工程が各場所で必要とする時間

A= 1h,B= 2h,C= 1h,D= 2h,E= 1h

(3)　各作業工程が並行して作業できるかどうかを与える作業ﾃｰﾌﾞﾙ

	A	B	C	D	E
A	X		X		
B		X			X
C	X		X		X
D				X	
E		X	X		X

これらの入力ﾃﾞｰﾀをもとにｼｽﾃﾑは(例えば)次の結果を出力する。

例(出力ﾃﾞｰﾀ)

各地点での作業工程表

	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
P1	A	C	B	B		D	D	E							
P2			D	D	B	B		A	C	E					
P3	D	D	A	C			E		B				B		
P4							A	D	D		B	B	C	E	

1．3．5　ルールの処理速度測定問題

名称

N個の自然数による繰り返しルール

目的

(1) ルールの個数・条件の個数と、ルールの条件照合に要する時間との関係を求める。

(2) 〃　ルールのコンパイル時間との関係を求める。

問題の定義

〔N個の自然数のうち連続昇順のM個の組合せを条件部に持ち、その結論部で条件部に含まれる先頭の自然数を含むデータを更新するN個のルールの組を実行せよ〕

留意事項

(1) ルールの条件部の指定で定数だけでなく、変数や構造化データを指定してもよい。また、自然数の替わりにAA、AB、AC等の連続昇順の文字列で記述してもよい。

(2) データの更新では条件部で照合する部分を更新しないこと。

(3) 処理の終了はルールのファイヤ数で1000、2000、4000を条件とする。

（ルールのファイヤ数＝1000×2^{j}，j＝1，2，3，・・・）

(4) 競合解決で選択される同時処理数が可変の場合、１０未満を指定すること。

(5) 通常N＝５０、M＝５程度を指定する。

記述の例

（条件部に変数を記述しない単純な構造のデータの場合N＝５０，M＝４）

```
(ルール１)      if      (1),(2),(3),(4)
                  then  modify  (1)
(ルール２)      if      (2),(3),(4),(5)
                  then  modify  (2)
    :
    :
(ルールｉ)      if      (i),(i+1),(i+2),(i+3)
                  then  modify  (i)
    :
    :
(ルール50)      if      (50),(1),(2),(3)
                  then  modify  (50)
```

50個のルールの組

(初期設定ルール)　if　　null

then　create　(1)
　　create　(2)
　　:　　N個
　　create　(N)

但し、上記の例は以下の規則で記述される。また、初期設定ルールは独立に処理する。

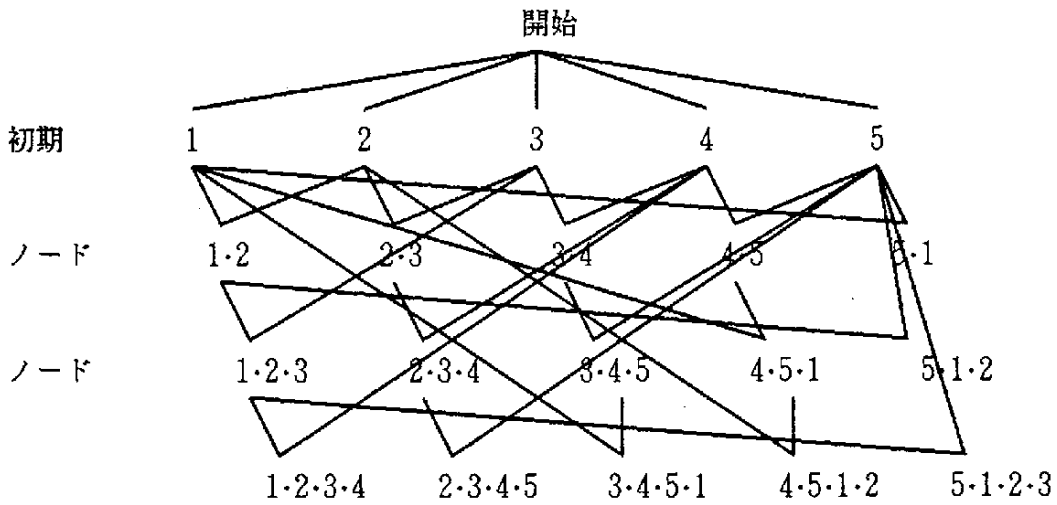
① (i),(j) はワーキング・メモリ内に整数 i を値として持つデータと、整数 j を値として持つデータとが共に存在する条件

② create (i) は整数 i を値として持つデータをワーキング・メモリ内に生成すること

③ modify (j) とは整数 j を値として持つデータをワーキング・メモリ内で更新すること（なお、照合対象データは構造を持ち、更新では照合用の整数 j は変更しない）

補足説明

(1) Rete-networkの構成の例（N＝5，M＝4の場合）

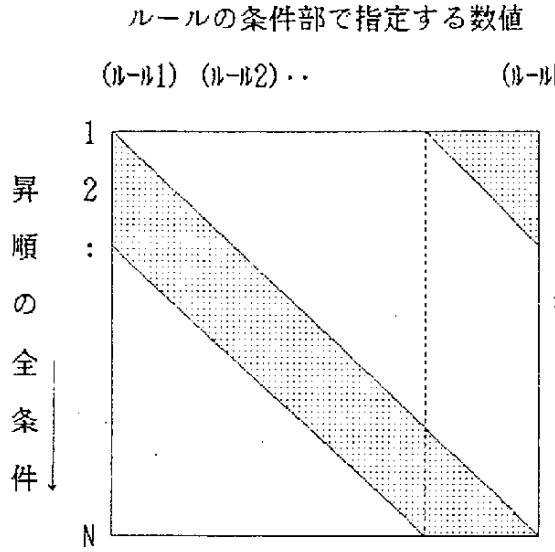


ルールの条件部･･･(ルール1)　(ルール2)　(ルール3)　(ルール4)　(ルール5)

(2) N個の自然数のうち任意のM個の組合せの全てを条件部に持つ場合も想定できるが、この場合のルール数は ${}_{N}C_{M}$ 個と非常に多くなるため注意が必要である。

(3) ハイブリッド型のツールでは、各自然数に対応するデータがフレームとして構造を持ち、かつ、それぞれがL個ずつ存在する場合をも実現できるが、通常はRete-networkは変わらない。

(4) 全条件とルールで指定する条件との関係は、以下の様な関係にある。



指定する条件は帯状に存在する

〔担当　桃井〕

1．3．6　実行性能に関する問題（フレーム関連）

名称

フレーム照合

目的

（1）システムに存在するフレームの個数と、フレーム照合に要する時間との関係を調べる。

（2）インヘリタンスを含んだ場合の照合に要する時間を測定し、（1）と比較する。

問題の定義

（1）問題1

① 0－99の整数を値としてとりうるスロット（スロット名をRand-numとする）を持つフレームを定義する。（レベル 0）

② ①で定義したフレームを親とする、0－99の整数乱数をRand-numスロットの値としてもつフレームをN個発生させる。（レベル 1）

③ ②で生成したフレームについて、与えられた整数をRand-numスロットの値としてもつようなフレームを全て検索し、検索に要した時間を測定する。

Nの値は500、1000、2000、4000とする。

（注 $N=500\times2^{M}$，M＝1，2，3，・・・）

（2）問題2

① (1)－②で発生させた各々のフレームの下に、Rand-numスロットの値を継承するフレームを発生させる。（レベル 2）

② ①で発生させたフレーム（レベル2のフレーム）について、(1)－③と同様に与えられた整数をRand-numスロットの値としてもつようなフレームを全て検索し、検索に要した時間を測定し、(1)と比較する。

（3）問題3

① (2)－①で発生させた各々のフレームの下に、Rand-numスロットの値を継承するフレームを発生させる。（レベル 3）

② ①で発生させたフレーム（レベル3のフレーム）について、(1)－③と同様にフレームを検索し、検索に要した時間を測定して、(1)・(2)と比較する。

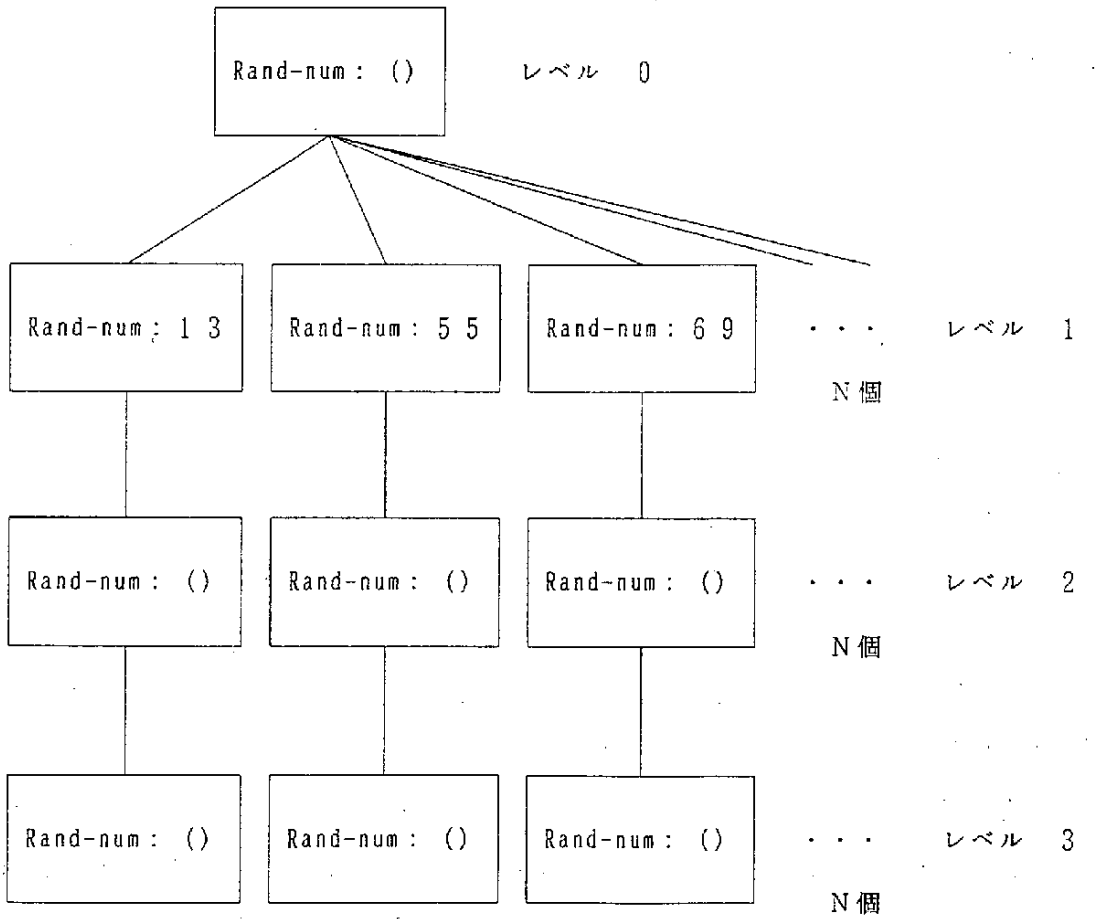
留意事項

（１）フレームの検索にはパターン・マッチ機能を用いること。

（２）性能評価に用いたツール／ハードウェアの組合せにより、Ｎの値は適当に増減させ、目的とする比較評価ができるようにする。

例

問題のフレーム構造



１．４．標準問題の背景

一般に、エキスパートシステムの開発においては、システムのスクラップ・アンド・ビルドを繰り返すラピッド・プロトタイピング、または、探査型プログラミングと呼ばれる方式が適当といわれている。それはシステムに専門知識という把握しにくいものを取込むために、開発当初から要求仕様や問題の解法を厳密に定めることが困難であり、このような手法によらなければ、システムの開発効率が悪くなるという事情による。また、実際のエキスパートシステム開発においては、問題解決の知識を確定するまでに、数多くの議論が必要な場合が多く、それには専門家ーインタビューー知識技術者という単純な構造以外のアプローチも必要となる。

ツールを評価する場合でも、このようなエキスパートシステムの特性を考慮することが望ましいが、このような開発方式・知識収集方式のもとで、同一の問題を用いた評価を実施することは事実上不可能である。そこで、我々は、現実にエキスパートシステムを用いて解かれている問題を簡略化し、比較的大規模な標準問題を、合成型・分析型について設定した。そして、それらを、代表的なツールによってインプリメントすることで各ツールの機能・性能を比較検討することとした。

２つの標準問題は、レンズの骨組み設計問題（合成型の問題）と電気設備のトラブルシューティング問題（分析型の問題）である。これらは、ともに現実の問題を大幅に簡略化してえられたもので、他の評価用問題に比較すると十分に複雑かつ大規模なものではあるものの、対象となる問題は明確に定義されているwell-definedな問題である。この２つの問題は、代表的なハイブリッド・ツールであるＫＥＥ，ＡＲＴ，Knowledge Craft(KC)の３つを用いてそれぞれインプリメントされ、あわせて６つのプロトタイプシステムが実現された。この内容については２章で詳しく解説されるので、ここでは、２つの標準問題が、もとの、現実に取り扱われている問題とどのような関係にあるかを解説し、これから標準問題を設定したことの妥当性について考察する。

レンズの骨組み設計問題（合成型の問題）[菊地 86]は、仕様決定から生産設計にいたるまでのレンズ設計工程のうち、従来専門家のはたす役割の大きかった部分のエキスパートシステム化をはかるものである。この段階では、方式設計で与えられたレンズの機能・性能にしたがって、レンズシステムの基本的な骨組みのパラメタを決定する。レンズシステムのパラメタは５０～１００個からなり、各レンズの面の曲率・ガラスの種類・厚み

・各レンズ間の間隔で構成される。骨組み設計では、制約を満たしながら、各レンズ群の焦点距離と群間の距離とを決定する。従来の設計作業では、専門家が膨大な数値計算プログラム（ＣＡＤ）を利用しながらシミュレーションを繰り返すことによって骨組みの設計を行っていた。実用に供されているのエキスパートシステムは、この作業を支援するもので、制約条件を満たしながら適切なＣＡＤコマンドを生成する機能をもつ。標準問題として設定したのは、このうち、対象をズームタイプレンズに限り、また、膨大なＣＡＤシステムをごく単純な数値計算プログラムに置き換えたものである。本システムは巨大な状態空間の中から制約条件を満たすパラメタの組を見出すという意味で、典型的な設計型問題であり、実際のエキスパートシステムと比較すると規模は小さいものの、人工知能技術の適用という立場からは本質的な部分をすべて含むので、標準問題として適当と判断した。

電気設備のトラブルシューティング問題（分析型の問題）[湯井 87] は、鉄鋼会社における設備故障診断問題のひとつを取り扱っている。特に製鉄所は２４時間操業を行っており、整備要員の不足する夜間の設備故障に対応する作業の効率化が必要とされている。さらに、電気設備の診断には経験的知識ばかりではなく、電気回路など設備の機能・構造に関する知識も必要であり、設備の改善に伴ってこの種の知識は常に変更される性格のものである。本システムは、現場で対象機器の測定を行う担当者が使用することを想定し、広範囲にわたるトラブルシューティング作業をオフラインで支援することを目的に検討された典型的な診断型エキスパートシステムである。本システムの特徴は担当者みずからが知識を投入・変更できるようにごく単純な知識表現を採用したこと、経験的知識と構造的知識の両方を診断に利用していること、宣言的な知識と手続き的な知識とを併用していることにある。標準問題では、このうち、対象設備をサイリスタレオナード装置にかぎり、そのＳＣＲゲート遮断が発生した場合の（直接的な）故障診断・原因究明のみを扱っている。この結果、本問題はフォールトツリーの探索を中心とする問題となっている。これは、診断型のエキスパートシステムではしばしば現れるタイプの問題であり、標準問題としてツールの評価に利用することは適当と考えられる。

参 考 文 献

[菊地 86] 菊地一成（他）：レンズ設計エキスパートシステム．情報処理学会知識工学と人工知能研究会，44-4，1986.

[湯井 87] 湯井勝彦，森谷正晴：鉄鋼業におけるエキスパートシステムの応用．電気学会論文誌C，Vol．107，No．2，pp．115-120（1982）．

寺野　隆雄（（財）電力中央研究所）

1．5．評価項目リストと標準問題の利用について

Clancy等[Clancy 87]は、AAAI 87 のチュートリアルでエキスパートシステム開発ツールの評価問題を論じ、評価方法として次の８つのステップを提案している。

①開発する問題の性質を定める。
②目標を適切なレベルに定める。
③ツールに必要な機能を定める。
④ツール評価の尺度と評価方法を定める。
⑤利用可能なツール群を選ぶ。
⑥ステップ③、④の結果を利用して候補を絞り込む。
⑦設定したフレームワークにおいて優先的な項目を決定する。
⑧ツールを評価し、選択する。

前節までに述べた評価項目と標準問題は、このうち、ステップ③～⑥までの作業を支援するのに有用なものである。そして、ツールの各評価項目については、ツールのベンダーが、テスト問題を利用した性能評価結果については、中立的な機関が内容を記述し、それらを利用者が自由に検索できるようなシステムを開発することが必要と考えられる。また、２つの標準問題についてはその厳密な仕様を公開して、知識処理技術やエキスパートシステムの教育時に利用することが考えられる。

参　考　文　献

[Harmon 85]]Harmon,P.,and King,D.: "Expert Systems -ArtificialIntelligence in

Business." John-Wiley,1985.

[日経 85] 日経エレクトロニクス：" 商品化相次ぐエキスパート・システム開発用ツールを比較する." 1985.11.4,pp.153-175.

[Gilmore 86] Gilmore,J.F., Pulaski,K., Howard,C.: A Comprehensive Evaluation of Expert System Tools. Proc.of SPIE,Vol.635 (Applications of Artificial Intelligence III), pp.2-16(April 1986).

[Richer 86] Richer, M.H.: An Evaluationof Expert System Development Tools. Expert Systems, Vol.3, No. 3, pp. 166-183 (July 1986).

[Wall 85] Wall, R. S., et al.: An Evaluation of Commercial Expert System Building Tools. Data & Knowledge Engineering, Vol. 1, No. 4 pp. 279-304 (1985).

[Beach 87] Beach, S.S.: "Analysis of High-End Expert Systems Tools." Spang Robinson Report, 1987.

[NASA 86] NASA: "Expert Systems Handbook." Johnson SpaceCenter Report, 86-FM-10, JSC-22230,July 1986.

[Faught 85] Faught,W.S.: "Functional Specifications for AI Software Tools for Electric Power Applications." EPRI Report NP-4141, Research Project 2582-1, August 1985.

[Chandrasekaran 1986] Chandrasekaran, B.: Generic Tasks in Knowledge-Based Reasoning: High-Level Building Blocks for Expert System Design. IEEE Expert, Vol.1,No.3,pp.23-30 (Fall 1986).

[小林 86] 小林重信：" 知識工学. " 昭晃堂, １９８６.

[Clancy 87] Clancy, W.J., and Rothenberg, J. "Evaluating Expert System Tools." Tutorial of the 6th AAAI (July,1987).

寺野　隆雄（（財）電力中央研究所）

2. 標準問題によるエキスパートシステム開発ツールの比較評価

2. 標準問題によるエキスパートシステム開発ツールの比較評価

はじめに

本章では，標準問題として二つの問題領域から例題を選定し，三種類の代表的なエキスパートシステム開発ツールで実際にインプリメントした結果について述べる。

二つの問題領域としては，設計問題と診断問題の領域を選定し，それぞれレンズの骨組み設計支援問題と電器設備のトラブルシューティング支援問題を取り上げた。各問題の概要については本章の付録1を参照されたい。これらの問題は，実際にもエキースパートシステムの対象としてシステム化されているものであり，ここではツール評価の標準問題として手頃な大きさになるように簡略化されている。標準問題設定の背景および妥当性については一章の1．4節に詳しくのべられている。三種類の代表的なツールとしては，汎用のハイブリット・ツールであるART,KEE,Knowledge Craft(KC) を用いた。ART と KEEは S Symbolics マシーン上のものを，KCは VAX11/780上のものを使用した為，開発環境やユーザインタフエース環境の評価については同様に扱えないことを注意する必要がある。ツールのバージョンについても，KEE では評価の作成中に2.1 から3.0 にアップした為，評価の一貫しない点もあるが，できるだけ最新のバージョンに統一するようにした。

実際のインプリメントについては，ツールごとに担当者を割り当て，各担当者が二つの問題を順にインプリメントする方式をとった。これは効率よくインプリメントを行ないたいと考えたとき，問題を理解するよりもツールの習得により時間がかかると判断したことによる。しかしこれは担当者の入れ替わりもあったため必ずしも厳密ではない。インプリメントの期間を十分取ることが出来る状況であれば，問題毎に担当者を割り当てそれぞれのツールでインプリメントする方式をとったほうがツールの評価をより公平に判断できたかもしれない。次節以降，2.1 節では，まず，二つの標準問題について知識処理システムに関する調査研究・報告書－知識システム開発方法論－の第3章に論じられている主旨に沿って分析を行ない，問題のクラス，特徴，基本タスク，必要とされる問題解決機能を明確にすることを試みている。2.2 節では，2.1 節の分析結果を受けて各問題の基本タスクと問題解決機能がそれぞれのツールでどのように記述されるのかをできるだけ具体例を示しながら説明している。2.3 節は，2.1 節で説明した各ツールによる記述をまとめ直しながらツール間でのおおよその比較・評価を試みている。 2.4節では，実際にインプリメントを行なってみて，開発に要した工数，作成の容易性，作成されたシステムの評価についてまとめている。2.5 節では各ツールの開発環境に関する評価を，マシン環境の違い等に注意しながら述べている。

2.1　エキスパート開発ツール評価問題の解析

付録１で与えられたエキスパート開発ツールの評価問題を類型的タスクの概念からこれらの問題を分析する。これによって，システム工学アプローチから，よりプログラミングレベルでの各ツールの比較を与える材料を提供する。

2.1.1　レンズの骨組み設計支援問題

2.1.1.1　問題の概要

レンズ設計の５つの工程の（仕様決定，方式設計，骨組み設計，詳細設計，生産設計）うち骨組み設計を対象問題とする。レンズの骨組み設計とはレンズ群の焦点距離や群間隔を機能及び形状寸法の制限の中で選定することである。骨組み設計は以下の４つの処理モジュールに分けて考える事が出来る。これらの処理をエキスパートシステムとしてインプリメントし，条件を充たす解を求めるシステムを構築することが本問題の目的である。問題の詳細については付録１を参照されたい。

① 仮説の初期パラメータを決定する。通常は仮説パラメータとしてシステムの与えるデフォルトの初期値を仮定し，最初の仮説とする。

② 仮説パラメータから骨組みパラメータおよび評価対象パラメータを直接計算式もしくは数値シミュレーションをおこなうことにより求める。

③ 評価対象パラメータの値が制約条件値を満たせば（原因分析ルールが１つも実行されない）その仮説は１つの解を与える。そうでなければ，原因分析ルールにより不具合原因が幾つか求められる。

④ 仮説更新ルールを用いて，仮説の補正をおこない新たな仮説を生成する。仮説更新ルールにより同時に複数個の仮説が生成されることがある。

2.1.1.2　問題の分析

2.1.1.2.1　問題のクラス

本問題は，合成型のクラス3（システムの構成があたえられており，構成要素の特性を選択し，決定する問題）に分類される。ただし，本問題では，システムの構造を，システム工学的な問題の階層的な認識から，つまり，与えられたシステム構造を適当な階層に対応させ，計画選択・精密化を各ノードにおける作業に対応させることからは，とらえていない。むしろより現実的に，条件と，その下での知りたい解の構造が明確になっていることを前提に問題解決を行っている。

本問題がこのように，合成型の問題としては容易になるのは，対象がレンズ設計の一連の流れの中のごく一部であり，レンズの骨組み設計はパラメータの精密化の1つにほかならない事とも関連する。しかし一方では，本システムが与える解は階層的に生成される。すなわち本問題が与えるシステムの問題解決の基本的なアプローチは解の生成と検査による手法（Generate and Test)に従っている。このことは，この種の設計問題を解くにはそれに陰に含まれる階層的な構造を利用することが重要なことを示している。これは，従来型の線形または非線形の計画問題で解の発見される過程と本問題における解の発見過程とには本質的な違いがあることを示している。

2.1.1.2.2　問題の特徴

一般的に設計問題は合成型問題としてとらえられる。この立場から，本問題の特徴をまとめると次のようになる。

分　　類	本　問　題　の　特　徴
1.システム構造の解空間が大	解空間はR^nのsubsetである。 （n：骨組みパラメータの数） ここでの解空間は（非線形）計画法における解空間と同じ構造である。解の数は一般に非可算無限である。
2.解析による合成が基本的	本システムでは，仮設空間の設定と解除を行なう

	ことにより実現している。基本的にGenerate and Testの方式で探索を行っている。
3.構成要素の最適化の必要性	本システムの解は最適解の候補を絞りこむための前段階と考えられるので最適化は行なっていない。
4.検証の安全性および，効率性	検証は各パラメータを初期値としたシミュレートを行ないその結果を検討することによりなされる。検証に必要なパラメータはそれを導出する関数か，またはシュミレーションプログラムを起動することによって与えられる。したがって，検証の完全性とは検証を受けるパラメータの選び方に依存し，効率性はシュミレートを起動させる順序などに依存する。
5.対話形式での設計支援問題	設計問題に限らず，エキスパートシステムでの対話形式の重要性すでに指摘されている。本システムでは，各種パラメータの変更をインタラクティブに実行出来るようにしている。
6.設計段階に応じた支援形態	本システムはレンズ設計の一部をシスムとして実現している。したがって，全体の設計段階の一形態として考えられる。

2.1.1.2.3　基本タスク

設計問題における基本タスクと本問題との関連は次のようにまとめられる。

基本タスク	本問題における表現
1.構成要素とその関連の表現	次の異なった立場での構成要素が考えられる。 レンズ設計全体をシステムとして考えた場合。 構成要素　＝（a）骨組み設計 Generate and Test の解空間を構成要素として見た場合。 構成要素　＝（b）仮説空間 設計対象を構成要素として考えた場合。 構成要素　＝（c）各レンズ群 各立場での各構成要素間の連結 （a）本題はレンズの骨組み設計のみを扱っているので他の処理との連結を表現する必要がない。 （b）各仮説空間の間の関係は，新しい仮設を産みだすことによる親子の関係，矛盾する2つの仮説が生じた時にその一方を消去するための無矛盾性の制御が対象になる。 （c）骨組みパラメータを構成する各要素を構成要素と考えると，それらの関係を表わす表現は各パラメータ間の評価式による関係や，各パラメータ値をもとにシミュレーションするこ

	とによって与えられるパラメータの検証などになる。これらの関係は陽に与えられるのではなく，むしろGenerate and Test におけるTestをおこなうパラメータを導出するときに陰に与えられる。
2.設計問題の階層的表現	本レンズ設計問題は対象問題を階層的に構造化していない。
3.代替案の自動生成	本システムの制御構造の基本的な部分である。骨組パラメータを導く仮説パラメータ群をルールを用いて自動生成する。新たな仮説を産み出すタイミングやどの仮説パラメータの修正に専門家の知識が組み込まれる。
4.部分システムの評価	本システムの部分システムとして考えられるのは仮説パラメータ，骨組みパラメータ，およびそれらから計算される評価，検証の対象となるパラメータ群である。また，各パラメータの評価は各パラメータ群が与えられた制限値の範囲内にあるかをチェックすることである。
5.上位レベルへの知的後戻り	本システムでは取り扱っていない。
6.設計事例の検索と利用	本問題では取り扱わない。
7.並列的問題解決	本問題では取り扱っていない。ただし，レンズ設計システムの全体をインプリメントするという立場にたてば取り扱う必要が出て来るであろう。

2.1.1.2.4　問題解決機能

レンズの骨組みパラメータの設計支援問題と問題解決機能との関連をとりまとめ。

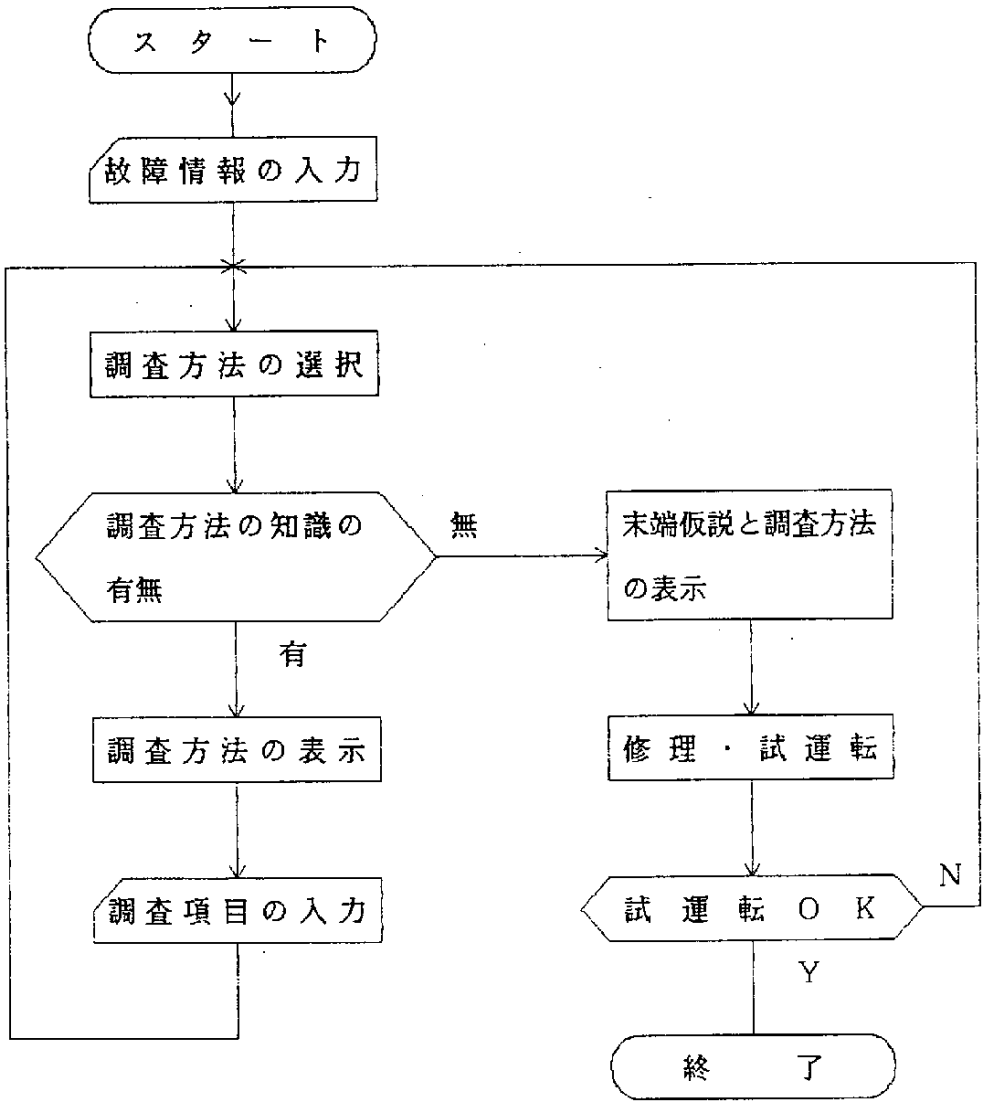
前述したように本問題はChandrasekaranの分類による設計問題で指摘されたシステム構造としての階層性を持っていない。したがって，それに関連した問題解決機能のすべてが本問題とマッチするとは考えられない。むしろ，本問題はGenerate and Test の適用に向いたシンプルな問題であると考えたほうが良い。さらに，トップダウン精密化戦略や拘束最小化戦略は必要としない。また，解の最適化の機能もここでは必要としない。本問題で必要となる解は，レンズ設計の以後の過程でさらに精密化され，検証をうけるからである。したがって，本問題の解はある程度のフィージビリティを満たしていればよい。また，協調型の推論構造の必然性がないことも前述したとおりである。したがって，本問題で関連する問題解決機能は，検証機能と仮説推論による知的探索機能の２つである。

2.1.2　電気設備のトラブルシューティング支援問題

2.1.2.1　問題の概要

対象としている問題の概要は次のとおりである。すなわち，電気設備（サイリスタレオナード装置）の保護装置が作動して電動機が停止し，ＳＣＲゲート遮断が発生した（パネルにそう表示された）場合に一連のトラブルシューティング作業（情報収集，計測機接続，原因追及，復旧，試運転）をＣＲＴとの対話形式でオフラインにて支援する。

本問題の処理は次の図のような手続き的な流れに要約される。



ここで調査方法の選択を支援する手法が３通り与えられている。それらは，テーブルタイプ，因果関係を与えるツリーの探索，および，手続き的な処理によるフロータイプに分けられる。上の図ではテーブルタイプによる選択が与えられているが，他の方法でも調査を細かくしていき，故障個所を絞り込んでいくという処理の流れは同じであり，その意味で，本問題はフォールトツリーの探索問題に帰着される。なお，本問題について詳しくは，

付録１を参照されたい。

2.1.2.2　問題の分析

2.1.2.2.1　問題のクラス

本問題は解析型，それも，典型的な診断型の問題である。すなわち，システムに異常または故障が生じた時，観測されたデータおよびシステムに関する因果関係または対象モデルの知識を利用して，原因を同定する作業をとり扱う問題である。

2.1.2.2.2　問題の特徴

分　　類	本問題における特徴
1.設計レベルの知識の利用	本問題は故障原因仮説およびそれを検証するための調査方法が階層的に与えられており，設計レベルの知識はそのツリー構造内に表現されている。
2.経験的知識の利用	経験的知識も上と同様に仮説，調査のツリー構造内に表現されてる。 本問題では経験的知識と設計的知識が必ずしも明確に分離されていない。
3.経験的知識の不確実性	仮説の評点を求めるときや，調査方法の評点を求めるときなどに使われる。仮説と原因を結ぶ曖昧性を表現するのに必要である。
4.対話的診断の推論制御	本問題は基本的には１，２であたえられた知識のツリーをフォールトツリーだとみなし，故障原因を探索していく対象システムが故障かどうかまた修正によりシステムが正常に起動したかどうかなどのチェックはすべてエンドユーザとの対話により決定さ

	れる。

2.1.2.2.3　基本クスク

分　　類	本問題における特徴
1.異常事態の分類層的表現	本問題では電気設備トラブルシューティング支援問題のうちＳＣＲゲートの遮断が発生した場合に問題を限定している。 従って，本問題では異常事態の範囲がすでに与えられている事を仮定している。
2.システム構造の階層的表現	本問題は基本的にはフォールトツリーの探索問題に帰着される。したがって，システム構造自身も階層的に表現されている。
3.解釈問題のタスクを内包	なし。
4.計測点の選択的決定	なし。
5.異常原因の同定	原因を探るのが本システムの目的である。
6.浅いモデルによる効率性	本問題では浅いモデルを扱っている。
7.深いモデルによる完全性	なし。

2.1.2.2.4　問題解決機能

本問題で必要な問題解決機能は，①仮説に従いそれを検証するという意味での目的駆動型推論，②仮説，調査方法の選択に必要な不確実性の伝達，③因果関係を表わすand/orツリーのパステストなどである。

2.2　各ツールによる基本タスクおよび問題解決機能の表現

2.2.1　レンズの骨組み設計

2.2.1.1　問題の特徴

2.2.1.1.1　Generate and Test

次の推論制御により実現される。即ち，仮説パラメータの設定→骨組みパラメータの計算→制約条件による評価→原因との対応→仮説パラメータの修正の順に実行される。

(1)　ＡＲＴ

コンテキストのsprout〔仮説空間の生成〕→仮説空間内に計算されたパラメータをassertする→ルール群の条件部による制御とそのプライオリティによる「制約条件による評価，原因の対応」の順序付け→新たな仮説空間のsprout。

(2)　ＫＥＥ

ワールドの生成〔仮説空間の生成〕→仮説空間内に計算されたパラメータをassertする→ルール群の条件部による制御とそのプライオリティによる制約条件による評価，原因との対応の順序付け→新たな仮説空間のcreate。

(3)　ＫＣ

仮説空間をそれが表現するスロット値で与える→ルールの条件部による制御で各処理の順序づける。

2.2.1.1.2　対話形式での設計支援

プログラムは各種パラメータの表示機能と変更機能を実現したコンテキストやワールドの表示は各ツールに備わっている機能を用いている。

(1)　ＡＲＴコンテキストの表示「各仮説空間の生成木の動的変化」，各種パラメータの表示編集

(2)　ＫＥＥワールドの表示「各仮説空間の生成木の動的変化」，各種パラメータの表示編集

(3)　ＫＣ各種パラメータの表示編集

2.2.1.2　基本タスク

2.2.1.2.1　構成要素とその関連の表現（仮説空間の表現）

(1)　ＡＲＴ：「コンテキストによる表現」

ＡＲＴではviewpoint 概念におけるコンテキストをもちいて定義している。新たなコンテキストを生成するタイミングは仮説パラメータの変更を行なった時である。コンテキスト間の自動マージング機能を制限することにより，各仮説空間のモジュール性を高めている。ＡＲＴには仮説世界を定義するデータ表現（hypothesize)があるが，本問題ではいちど立てた仮説は基本的に変更させないため本問題では使いづらい。

(2)　ＫＥＥ：「ワールドによる表現」

ＫＥＥでは，仮説空間を１つのWorld(ＫＥＥ World）に対応づけている。新たな，仮説(World)を生成するタイミングは，仮説パラメータの変更をおこなったときである。ＡＲＴと同様に，World の自動マージングは生じないようにしている。ＫＥＥには，ＡＲＴのHypothesize の概念はない。

(3)　ＫＣ：「スロット値による分類」

ＡＲＴ／ＫＥＥ版との相違は，各スキーマにid（identification）スロットを持たせ，仮説空間の判別をさせたことである。従って，新しい仮説空間で継承の必要なパラメータ

はすべて新しくmakeする必要がある。

2.2.1.2.2　各レンズ群の表現

レンズ群を含む骨組みパラメータの表現は各ツールともフレームで与えられている。

ＡＲＴ（スキーマによる）

```
(defschema honegumi
        (f1)
        (f2)
        (f3)
        (f4)
        (f5))
```

ＫＥＥ（ユニットによる）

```
(HONEGUMI                                ファセット
        (F1 NIL ((MEMBERP NUMBER.OR.EMP))..)
        (F2 NIL ((MEMBERP NUMBER.OR.EMP))..)
        (F3 NIL ((MEMBERP NUMBER.OR.EMP))..)
        (F4 NIL ((MEMBERP NUMBER.OR.EMP))..)
        (F5 NIL ((MEMBERP NUMBER.OR.EMP))..)
```

ＫＣ（スキーマによるがＫＣではスキーマ文以外にリテラライズを宣言する必要がある）

```
(defschema honegumi
        (f1)
        (f2)
        (f3)
        (f4)
        (f5))
```

```
(schema-literalize honegumi
        f1
        f2
        f3
        f4
        f5)
```

2.2.1.3　代替え案の自動生成（仮説パラメータの修正）

その時点で処理の対象となっているコンテキスト（view point）から，新たにコンテキストを生成する部分のルール記述を与える。

(1)　ＡＲＴ　仮説空間の生成

「コンテキストの生成」

```
(defrule kasetu-kousin
    (viewpoint  ?v
          (schema kasetu
                (elw ?elw)
                (elw ?elw)
      .
      .
          (schema tansaku
                (delw ?delw)
      .
      .
⇨
(at ?v
 (sprout
  (modify
   (schema kasetu
    (elw =(plus ?elw (minus ?delw))
        )))))
```

(2)　ＫＥＥ　仮説空間の生成

「ワールドの生成」

```
(IF (THE E1W OF KASETU IS ?E1W)
      .
      .
      .
      .
      .
      .
(THE DE1W OF KASETU IS ?DE1W)
      .
      .
THEN
IN.NEW.WORLD
(CHANGE.TO
(THE E1W OF KASETU IS (+ ?E1W(-?DE1W
 ))))
```

2.2.1.4　部分システムの評価

制約条件により，評価の対象となるパラメータをチェックし，その結果その仮説空間を削除するか，対応する原因フラグをたてるなどの処理をする。評価の対象となるパラメータを導出する部分はリスプ関数により記述している。対応する原因フラグを立てる部分はルールの実行部に記述する。これらの部分の処理には各ツールによる差はない。ツールに

よる差異は仮設空間を削除する部分で生ずる。ＫＣによる版は単純にその仮説空間をｉｄスロットに持つスキーマをすべて削除することにより実現している。ＡＲＴ及びＫＥＥによる仮説空間の削除について以下に述べる。

(1) ＡＲＴ (poisonによるコンテキストの削除)

```
(defrule poison-rule
        (viewpoint ?v
           (schema hyoka-para
                  (es2amp ?es2amp)
                  .
                  .
           (schema seigen
                  (es2amp ?es2ampp)
                  .
                  .                    ←リスプ関数
  (test (> ?es2amp ?es2ampp))
  ⇨
  (at ?v                  ←名前は何でも良い
      (poison "dan")))
```

(2) ＫＥＥ (deduceによるワールドの削除)

```
(IF (THE ES2AMP OF HYOKA IS ?ES2AMP)
    (THE ES2AMP OF HYOKA-SEIGEN IS ?ES2AMPP)
    (LISP (> ?ES2AMP ?ES2AMPP))
    THEN
    DEDUCE                          ※ KEEのルールのタイプは，new-worle action/
                                       reduction/same-world acton があり，DEDUCE
                                        はreduction ルールに属し，最優先される
    FALSE))
```

2.2.1.3　問題解決機能

2.2.1.3.1　検証機能

解の検証による枝刈りである。上のPOISON/FAULSE 文を参照してほしい。

2.2.1.3.2　仮説推論およびそれによる知的探索機能

評価関数の使用による知的探索機能であるが，コンテキストとの関係も有りうまくインプリメントされていない。

2.2.2　電気設備のトラブルシューティング支援問題

2.2.2.1　問題の特徴

2.2.2.1.1　設計レベルの知識および経験的知識の利用

これらの２つの知識は専門家の知識としてそれぞれ次のデータ表現によって実現されている。

フォールトツリーの表現
仮説と調査項目に関する知識
調査の容易性に関する知識
調査方法に関する知識
因果関係に関する知識

これらの知識の表現は各ツールともフレーム（スキーマ）で表現している。ただし，ＫＥＥ，ＡＲＴによる表現とＫＣによる表現は若干異なっている。これはツールのもつ表現力の違いではなくインプリメント手法の違いからきている。

(1) フォールトツリーの表現

① ART

ツリー中の各ノードをそれぞれ1つのスキーマで表現し，そのスキーマの中に自分の親を表わすスロットと，子を表わすスロットを持つ。フォルトツリーのノードには，仮説ノードと，調査方法ノードの2種類がある。各ノードは，この2種類のノードのインスタンスとなっている。

② KEE

ARTの構造と同じであるが，調査方法ノードをフロータイプ，リーフタイプ，テーブルタイプおよび因果関係を表わすノードにクラスとしてまとめている。

③ KC

複数のノードをまとめて管理するために，フォールトツリーの各分岐に対応させて，その分岐に与えられた仮説を導くために必要となる知識を管理するスキーマを対応させている。

(2) 仮説と調査項目に関する知識

① ART／KEE

仮説ノードスキーマのcorrelation スロット値として持つ。

② KC

フォールトツリーを与えるスキーマ内でsymptom-schemaのスロット値としてスキーマとしてもつ。

(3) 調査の容易性に関する知識

① ART

調査方法スキーマのtotal-accessmentスロット値として持つ。

② KEE

ARTとの相違は容易性等の相乗平均が必要になった場合if needed デモンで計算するようにしている。

③ KC

調査方法スキーマ内のスロット値としてもつ。

(4) 調査方法に関する知識

① ＡＲＴ

調査方法スキーマ内のスロットprocedure にスロット値として，調査を行なうためのＬＩＳＰ関数名が書いてある。ルールの実行部でfuncall で呼びだす事によりこの関数に起動がかかる。

② ＫＥＥ

procedure スロットにメソッドとして書かれている。起動はprocedure スロットにメッセージをおくればよい。

③ ＫＣ

ＫＥＥと同様に，調査方法スキーマ内の対応するスロットにメソッドで書かれている。この呼び出しはcall-method による。

(5) 因果関係に関する知識

① ＡＲＴ

後ろ向きルールで記述している。or結合では各アークに対応してルールを記述し，and結合に関してはまとめて表現する。

② ＫＥＥ

後ろ向きルールでの記述である。Ver.2.0 での記述であるために，ＡＲＴのようにすなおには表現できていない。後ろ向き推論の起動にはＱＵＥＲＹ文をもちいる。ＱＵＥＲＹ文のパラメータのaskuser フラグをオンモードにすることによってtell and askの条件部を自動的にユーザへ質問する。

③ ＫＣ

因果関係もフォールトツリーの特別な場合としてスキーマで表現する。ツリー上の探索はＬＩＳＰ関数で記述している。

(6) 経験的知識の不確実性

この特徴を表わす知識は上記の調査の容易性に関する知識に含まれている。上記を参照してほしい。

(7) 対話診断の推論制御

推論制御はフォールトツリーの探索法により与えられる。各ノード上での調査した結果をユーザが答える事によって異常の原因を同定する。

① ART

KEE／KCと異なり前向きのルールを用いて推論制御を行なっている。フォルトツリーの探索問題のようなかなり手つづき的な制御をプロダクションルールで表現するのはやりづらい部分がある。次に探索するノードを与えるために，ルートコンテキストに次の制御をおこなうノード名を与え，これによって，次に発火するルールをコントロールする。

② KEE

オブジェクト指向プログラミング的に書かれている。フォールトツリー上の調査方法ノードのなかに，メソッドがかかれており，このメソッドの間にメッセージが行きかうことで，推論が進んで行く。

このうち，フォールツリーを降りるためのメソッドは，DOWN，FAULT，TREEメソッドであり，各調査方法のタイプにより4通りである。

③ KC

全体の推論制御はLisp関数で記述している。フォールトツリーの探索は各調査ノードが対応するhas-assumption ノードに与えられえている子ノードを探索する。

2.2.2.2 基本タスク

本問題の基本タスクとしては，システム構造の階層的表現と異常原因の同定である。システム構造の階層的表現は，データ構造としてはフォールトツリーの知識の表現，各階層的な構造間の関連を表わす知識としては，フォルトツリーを探索する機能および，各ノードの調査実行機能に表現されている。

異常原因の同定を与える機能としてもフォルトツリーの探索機能および，各ノードの調査実行機能に表現される。

2.2.2.2.1　システム構造の階層的表現および異常原因の同定

フォールトツリーを探索する機能は上記のとおりである。各ノードの調査実行機能に関して以下に各ツールでの表現を与える。

(1)　ＡＲＴ

・テーブルタイプ　前向きルールの実行部にリスプ関数で調査項目をユーザに質問したり，次の調査方法を優先順に表示してユーザに選ばせたりする。

・因果タイプ　前向きルールの条件部に因果関係を表わす後ろ向きのルール群のルールの条件を記載し，対応する後ろ向きのルールを発火させる。

・フロータイプ　前向きルールの実行部にfuncall 文で対応するノードを与えるスキーマに書かれているリスプの手続き関数を起動させることにより実現している。

・リーフタイプ　ルールの実行部に直接記載する。そのリーフが故障原因ではなかった時には制御をルートに戻す。

(2)　ＫＥＥ

因果タイプをのぞき，一連のメソッドに順次メッセージを送ることで，そのノードについての調査を行なっている。ツリー探索中に，調査ずみのノードに再び，制御が来た場合は，調査済みである事を表示して，２度同じ調査は行わないようにしいてる。

因果タイプでは，因果関係ツリーのルートノードの正常・異常を後ろ向き推論で調べている。

(3)　ＫＣ

・テーブルタイプ・・・リスプ関数test-managerにより調査方法の選択と実行を管理している。優先度と調査方法の表示，調査方法の選択と実行などリスプ関数で書かれている。

・因果タイプ・・・これもリスプ関数を再帰的に呼びだすことにより実現しているand or ツリーのデータ表現はスキーマで表現している。

・フロータイプ／リーフタイプ・・・対応するhas-assumptionノードに継承されているflow-separation スキーマ内のメソッドを起動することにより実行される。

2.2.2.3　問題解決機能

本システムで問題解決機能として必要な機能は目的型推論，不確実性の同定および因果関係を与えるツリーのパステストだが，それらの機能はすでに問題の特徴および，基本タスクの節で論じた。すなわち，目的型推論についてはフォルトツリーの探索，不確実性の同定についてはテーブルタイプの調査ノードのデータ表現と実行機能，ツリーの全パステストについては因果タイプの調査ノードのデータ表現と実行機能をそれぞれ参照されたい。

2.3　各ツールのプログラム記述レベルでの評価

要　旨

本節では，プログラム記述レベルでの評価を各問題ごとに与えた評価項目に従ってまとめる。

2.3.1　レンズの骨組み設計支援問題

本問題をインプリメントするに当たりプログラム記述レベルでの評価項目として次の項目を与えた。

・データ表現の記述

・仮説空間の記述

・プロダクションルールと手続き関数とのインタフェース

・説明機能の記述

これらの項目に関する各ツールの具体的な表現手法は付録２で与えられる。本節では付録の内容をまとめ各ツールの比較を行なう。

2.3.1.1　データ表現の記述

ＡＲＴ，ＫＥＥ，ＫＣともにフレーム表現により各種パラメータを表現している。表現の方法に差はない。各ツールの持つフレーム表現の記述能力はかなり異なる「付録４を参照」。ただし，本問題のデータ記述ではこれらの問題による差を与えるほど複雑な関係を必要としていない。

評　価　　ＡＲＴ＝ＫＥＥ＝ＫＣ

2.3.1.2　仮説空間の記述

ＡＲＴとＫＥＥ（Ver.3.0）による版はコンテキスト機能を用いて各仮説空間を表現している。それにたいして，ＫＣによる版ではコンテキスト機能を用いていない。したがって，各仮説空間のモジュール管理および，データの継承などは開発者が管理をしなければならない。

ただし，本問題で仮説空間をコンテキストで制御する必然性はあまりない。これはデータ構造のモジュール化に役立つのは明らかであるがＫＣのようにプログラムレベルで管理

してもそう繁雑にならない。本問題のように構造的にシンプルな問題を取り扱う場合にあまりこった手法をとるとかえって記述が面倒になるのはよくあることである。

ＫＥＥとＡＲＴのコンテキスト機能の相違が若干ある。ＡＲＴの場合は多重世界を記述することができる。そのために１つのコンテキストの世界だけではなく複数の世界の重なり合った世界を表現できるが，ＫＥＥのワールドは１重の世界だけの表現である。ただし，本問題ではその多重のビューポイント機能は必要ではない。ＫＣのVer3.0ではＫＥＥの機能程度のコンテキスト機能が備わっている。

ただし，ＫＥＥのVer2.0版でもルール部の記述能力が弱かったことなどから考えるとversion の違いによる差は大きい。従って，ＫＥＥ，ＫＣ，ＡＲＴをまとめて評価することは難しい。

2.3.1.3　プロダクションルールと手続き関数とのインタフェース

ＡＲＴ（Ver.2.0)ではデモンの使用が出来ないがルール記述の中でリスプ関数の呼び出しや，入出力パラメータの受けわたしは自由にできる。またスキーマ内で手続き関数を表現することも可能である。

ただし，ＡＲＴでのリスト型の表現はsequenceで与えられており，これは，リスプにおけるリスト概念とは異なる。したがってリスプ関数に引き数を受けわたす時などsequenceとlistとの型変換が必要になる。

ＫＣでのプログクションからのリスプ関数の記述方法もＡＲＴと同様にそのままリスプ関数名を与える事により起動をかける事が出来る。ただし，ＫＣでのリスプ関数とのインタフェースはスキーマ内での手続き関数の記述や，デーモンの記述にある。

ＫＥＥでもＡＲＴ，ＫＣと同様に，ルール記述の中にlispオペレータを用いることによってリスプ関数を自由に書ける。ＬＩＳＰ関数のリターン値を変数にユニファイさせるためには‘=’をもちいる。

評　価　　ＡＲＴ＝ＫＣ＝ＫＥＥ

2.3.1.4　説明機能の記述

ＡＲＴ，ＫＥＥ，ＫＣともに別な説明機能を利用者側でプログラミングしてはいない。したがって，３節で触れたユーザインタフェイスとして各ツールに備わった説明機能を評価することとする。ＫＣはＫＥＥ，ＡＲＴとことなりＶＡＸの端末からインプリメントされた。従って，システムがあたえるユーザインタフェイスは他のツールと比べてかなり差が生じている。

そこでＫＣについては，特に触れないこととする。

ＫＥＥの説明機能は豊富である，例えばつぎのようなものがある。

(a)　なぜ，そのworld が削除されたか（false になったか）

(b)　world とworld のデータの比較

(c)　どの様にその事実が導かれたか

ＡＲＴの説明機能

データベース内のファクトに注目し，そのファクトが生成された経緯を詳しく調べる事が出来る。上の(c)に対応している。

2.3.2　電気設備のトラブルシューティング支援問題

本問題をインプリメントするに当たりプログラム記述レベルでの評価項目として次の項目を与えた。

・データ表現の記述

・推論構造の記述

・説明機能の記述

これらの項目に関する各ツールの具体的な表現手法は付録２で与えられる。本節では付録２の内容をもとに，プログラム記述レベルでの評価をとりまとめる。

2.3.2.1　データ表現の記述

因果関係を表わす知識を除いて，基本的に各ツールともフレーム（スキーマ）で記述している。因果関係を表わすツリー構造はＫＣではスキーマで表現しているが，ＫＥＥ／ＡＲＴでは後ろ向きのルールを用いている。スキーマの記述能力は各ツールで若干異なっている。付録の機能表および，マニュアルレベルでの評価に記述しているように，ＫＣの記述能力が最もすぐれている。オブジェクト風のメソッド，メッセージの受け渡しなどはＫＥＥが記述しやすい。ＡＲＴではデモンをサポートしていないため余分にデータを持たせなければならなくなった。（ＡＲＴのVer3ではある程度サポートしている。）

後ろ向きルールの記述も含めて，ルール記述能力はＡＲＴが優れているがこれに関してもＫＥＥ，ＫＣのニューヴァージョンではかなり充実している。

本システムで使用したフレーム表現の記述能力の評価　ＫＣ＝ＫＥＥ＞ＡＲＴ

本システムで使用した後ろ向きルールの記述能力の評価　ＡＲＴ＞ＫＥＥ

2.3.2.2　推論機能の記述

本システムで用いた推論機能は基本的にフォールトツリーの探索であるが，これをより細かくいえば，次のようになる。

フォールトツリーを下にたどる機能

フォールトツリーを上にたどる機能

フォールトツリーのノードにある情報を操作する機能

これらの機能を実現するのに，ＡＲＴでは前向きのルールによる記述をし，ＫＥＥではオブジェクト指向による記述を行っている。ＫＣはＫＥＥとにているが，若干のデータ構造が異なり，全体の制御をリスプ関数にまかせている。

ＡＲＴ（Ver.2)ではオブジェクト風のプログラム記述をすることはできない。ＫＥＥ，ＫＣでは，ルール記述が可能であるが，本問題をルールで表現する必然性はない。むしろ本問題はより手続き的に記述されたほうが表現しやすい問題である。従って，これらの推論機能による評価とは与えられた問題とプロダクションシステム，オブジェクト指向プログラミング，手続きによる各インプリメント手法との相性の問題と思える。

評　価　　ＫＥＥ＝ＫＣ＞ＡＲＴ

2.3.2.3　説明機能の記述

本問題での説明機能とはwhy，how，what-if などの説明付加機能，フォールトツリーの表示機能，因果関係ツリーの表示，テーブルの表示，調査方法の手続きのフロー図の表示などを意味する。

本システムで与えた説明機能としては，ＫＥＥではツリー構造を表示する関数を用いて各ツリー構造の表示ができる。他のツールではこの種の機能は実現していない。why，how などの機能は，ＡＲＴ，ＫＥＥではデバック用のエディタを用いることで代用できるが，これはエンドユーザ向きではない。ＫＣは他のツールとことなりＶＡＸの端末上でインプリメントされた。従って，ユーザインタフェースに関しては同じレベルでの評価はできない。

評　価　　ＫＥＥ＞ＡＲＴ

2.4　各ツールでのインプリメンテション結果による比較

2.4.1　レンズの骨組み設計支援問題

2.4.1.1　開発工数

各ツールを用いて本問題をインプリメントするのに要した工数を表2.4－1に示す。ただし，総時間数は，ツールのマニュアルレベルの学習から，実際のシステムが完成するまでに要した合計時間である。また，項目ごとの工数は総時間数を 100としたときの百分率である。

この問題は，最初にＡＲＴ (Ver 2.0)にインプリメントされて，その後ＫＥＥ (Ver 3.0)とＫＣ (Ver 3.0)に順次インプリメントされた。したがって，ＡＲＴ版では，他のツールと違い基本設計及びプロトタイプを作成する必要があった。ここで，プロトタイプは，本問題の基本的な制御の流れを保ち，ルール数及び関　数を非常にシンプルにしたものを考えた。プロトタイプを作ることにより，
context 機能の用い方や，hypothesize の欠点などの基本設計にかかわる重要な点が明確になった。ＫＥＥではほぼＡＲＴの設計に基づいて，インプリメント作業をすすめたので，設計には工数を要していない。

また，この問題ではシミュレーションのための数値計算が必要であり，そのために最初のＡＲＴ版では数値計算用のLisp関数の作成に80時間程度の工数を要した。ただし，この工数は表の工数のなかには入っていない。幸い，他のツールもLispベースのツールであったために，ＡＲＴ用のそれらのLisp関数はそのままＫＥＥとＫＣでも使用できた。

表2.4－1　開発工数

ツール / 項　目	ＡＲＴ	ＫＥＥ	ＫＣ
ツールの学習	53 (%)	63 (%)	55 (%)
設計（プロトタイプの作成）	15 (%)	—	16 (%)
プログラムの作成	15 (%)	14 (%)	10 (%)
デバッグ	17 (%)	23 (%)	19 (%)
総　時　間　数	520 (hrs)	440 (hrs)	505 (hrs)

2.4.1.2　作成の容易さ

問題の概要から明らかなように，インプリメント上の１つのポイントは，仮説を用いた推論をどのように実現するかという点である。したがって，ここでは作成の容易さを比較するためのめやすとして，仮説を用いた推論部分（仮説の生成と検証）の作りやすさを取り上げる。

ＡＲＴでは仮説に基づく推論機能（ＡＴＭＳ）が提供されているが，ＫＥＥとＫＣには直接そのような機能はサポートされていない。ただし，本問題では，ＡＲＴの仮説推論は使いづらい。ＡＲＴとＫＥＥでは，context 機能を用いて，各仮説空間のモジュール性を高めた。ＫＣの場合はcontext 機能を用いれば，仮説空間が実現可能と思われるが，今回のインプリメンテーションでは，context は用いていない。ＫＣの場合はすべて，作成者がデータの管理をしている。

ルールは，ＡＲＴとＫＥＥともに仕様書の表現をツールのルール表現形式にそのまま変換するだけで，容易に記述できた。ＫＣの場合はＣＲＬ－ＯＰＳを用いてルールを記述したが，その形式はＯＰＳ－５に準じている。

本問題に対する，いくつかの項目の作成の容易さの比較を表2.4－2に示す。

表2.4－2　作成の容易さ

ツール / 項　目	ＡＲＴ	ＫＥＥ	ＫＣ
仮説空間の実現	容　易	容　易	直接はサポートせず
推論部の実現（仮説の生成と検証）	Viewpoint の機能が豊富な分だけ難しい	World の機能がシンプルな分だけ簡単	

2.4.1.3　作成されたシステムの機能の評価

(1)　基本機能

ＡＲＴの場合は，基本機能はほぼ満たしているが，Viewpoint (merging nil) の深さがあるレベルまで深くなるとシステムダウンする。

ＫＥＥについても，基本機能はほぼインプリメントできたが，横型探索，縦型探索とも

に，世界の深さがある一定のレベルまで深くなるシステムダウンする。

ＫＣについては，基本機能はほぼ作成できた。

(2) 拡張機能

ＡＲＴ，ＫＥＥ，ＫＣともに評価関数の導入は不十分である。

また，入力データの表示，変更等の機能は，ＫＥＥではＫＥＥの機能をそのまま使っている。特に，ＫＥＥの場合はインタープリタであるという点から，任意の時点で，制限パラメータ，探索パラメータの値を変更できる。

ＡＲＴの場合は，ルール記述により入力データの表示，変更等の機能を実現している。

(3) 評　価

ＡＲＴとＫＥＥについては，上で述べたように推論の実行途中でシステムダウンする（ツールのバグと思われる）ために評価の対象とならない。

2.4.2　電気設備のトラブルシューティング支援問題

2.4.2.1　開発工数

各ツールを用いて本問題をインプリメントするのに要した工数を表2.4－3に示す。ただし，総時間数は，ツールのマニュアルレベルの学習から，実際のシステムが完成するまでに要した合計時間である。また，項目ごとの工数は総時間数を 100としたときの百分率である。

この問題は，最初にＫＥＥ (Ver 2.0)上にインプリメントされて，その後ＡＲＴ (Ver 3.0)とＫＣ (Ver 3.0)に順次インプリメントされた。レンズの骨組み設計支援問題と異なり，特にプロトタイプの作成は行わなかった。基本的には，最初にインプリメントされたＫＥＥに対する設計に基づいて他のツールにもインプリメントされた。

表2.4－3　開発工数

項目＼ツール	ART	KEE	KC
ツールの学習	60 (%)	54 (%)	60 (%)
設　計	20 (%)	20 (%)	10 (%)
プログラムの作成	18 (%)	20 (%)	18 (%)
デバッグ	12 (%)	6 (%)	12 (%)
総　時　間　数	460 (hrs)	515 (hrs)	440 (hrs)

2.4.2.2　システム作成の容易さ

問題の概要から明らかなように，これはルールベースのシステムではなくて，フォールトツリーの探索システムである。したがって，インプリメント上の1つのポイントは，ツリーの探索手続きをどのように記述するかである。ここでは，システム作成の容易さの比較のめやすとして，探索手続き部分の作りやすさを取り上げる。

本問題に対するいくつかの項目について，作成の容易さを表2.4－4に示す。

KEEの場合は，手続き部分は，オブジェクト指向プログラミング機能のメソッドですべて記述した。

ARTの場合は，手続き部分はルールで作成はしたものの，不自然さがある。これは，ルールはデータ駆動で発火するのが自然な姿であるが，この問題では，次に発火すべきルールを制御するために，わざわざ xxx-phaseなどの述語をアサートしたり，リトラクトしたりして，ルールを発火させるためのデータを作っているからである。

KCの場合は，全体の制御を含めて，Lispとオブジェクト指向プログラミング機能のメソッドで記述した。Lispの世界からスキーマのデータを自由に参照できるので，全体の制御はLispで自然に記述できる。KCでは，前向き推論にCRL－OPSを使うことができるが，それを推論制御部に使っていたら，ARTのルールによる制御と同様に不自然な小細工をせざるを得なかったであろう。

表2.4－4　作成の容易さ

項目＼ツール	KEE	ART	KC
フォールトツリーの表現	容易(unit)	容易(schema)	容易(schema)
推論部の実現 （フォールトツリーの探索）	容易(ﾒｿｯﾄﾞ)	困難(ﾙｰﾙ)	容易 (lispﾒｿｯﾄﾞ)
リスプ関数との インターフェース	容易	普通	容易

2.4.2.3　作成されたシステムの機能の評価

(1)　基本機能

各システムとも，問題の概要で示された基本機能はほぼ実現されている。各ツール上での基本機能の具体的な実現方法については，表2.4－5に示す。これは，手続き型の問題をインプリメントするときの，ツールの特徴の一端を反映している。

ＡＲＴでは，オブジェクト指向プログラミング機能はないので，ソメッドによる手続きの記述はできない。したがって，ＡＲＴはこの種の問題にはやや不利と思われる。

(2)　付加機能

付加機能については，ＫＥＥの場合は，フォールトツリーの表示をＫＥＥの関数を呼ぶだけで簡単に実現できたが，ＡＲＴの場合はツールの機能としてそのような機能が提供されていないので実現していない。ＫＣについては，ＶＡＸのキャラクター端末であるという制約のために，もともとグラフィック機能は使えないのでツリーの表示機能は実現していない。

(3)　評　価

ＫＣ＝ＫＥＥ＞ＡＲＴ

表2.4－5　機能の実現方法

ツール / 機　能	KEE	ART	KC
ツリー探索	メソッド	前向きルール	ＬＩＳＰ関数
テーブルタイプの調査	メソッド	前向きルール ＬＩＳＰ関数	ＬＩＳＰ関数
フロータイプの調査	メソッド	ＬＩＳＰ関数 前向きルール	メソッド ＬＩＳＰ関数
因果タイプの調査	後向きルール	後向きルール	ＬＩＳＰ関数
リーフの調査	メソッド	前向きルール ＬＩＳＰ関数	メソッド ＬＩＳＰ関数

2.5 各ツールの開発環境の評価

この節では，各ツールの上に知識ベースを構築する際に重要となるエディタ及びデバッグ機能について述べる。

ＫＥＥ及びＡＲＴはSymbolics 3600上で評価したので，ツールの機能が十分に発揮されているが，ＫＣはＶＡＸ－780 ／ＶＭＳのＶＴ100 端末での評価であるので，ＫＣの機能は 100%発揮されていないことをことわっておく。

2.5.1 ＡＲＴ

(1) エディタ

ＡＲＴ独自のエディタはなく，ＯＳの提供するエディタを用いる。Symbolics ではZmacs エディタを用いる。名前からわかるように，これはEmacs 系のエディタである。任意の時点でZmacs エディタを呼び出せるので，わずらわしさはない。

ファクト，リレーション，スキーマ，ルールは，どれも一個の定義文を最小単位として，Zmacs エディタの中から部分コンパイル（インクリメンタルコンパイル）できる。部分コンパイルした部分は，既存の実行環境とリンクされるので，編集→テスト→デバッグのサイクルがスムーズに行なえる。

(2) インタラクティブな編集機能

ＫＥＥのように，マウスを使ってスロットやスキーマをポインティングして，スロット値を変更したり，インスタンスを生成させるような機能はない。

(3) 整合性のチェック

シンタックスレベルでのエラーを検出するために，コンパイル及びインクリメンタルコンパイル時に次のチェックが行なわれる。

① 未定義スロット，スキーマの指摘
② スロット及びリレーションの引数の数
③ リスプ関数の引数の数
④ スキーマ，ルール，リレーションの重複定義の指摘
⑤ 未参照変数の指摘
⑥ ルール条件部とデータとの整合性チェック

(4) デバッグ機能

多様な機能を持つ。主なものをいくつか示す。

① ブレークポイント

browser → rule →（ルール名）→ break

で，指定されたルールが発火する直前でbreak がかかる。なお，上の→はメニューに従って，このように選んでいくことを示している。マウスでピックしてもよいし，キーボードから入力してもよい。

② 発火したルールの表示。

watch → rule で発火したルール名が表示される。

③ 実行されたルールと照合がとれた事実の表示。

watch → fact

④ なぜその事実

browser → fact → justification

で，この事実をjustify したルールと事実を表示する。

⑤ ある事実によりインスタンスされたルールの表示。

browser → fact → matches

⑥ ルール条件部と事実との照合状況

browser → rule → matches

で，ルール条件部と事実の現時点での照合状況を表示する。

⑦ 事実の影響

browser → rule → matches

で，この事実が照合して，インスタンス化したルールを表示する。

⑧ viewpoint 外部からの検索

viewpoint →（viewpoint 名）→ query →（fact記述）

で，任意のviewpoint に特定の事実があるか確認できる。

⑨ アジェンダの表示

watch → agenda

(5) ヘルプ機能

(4)のデバッグ機能には，オンラインヘルプ機能が付いておりマニュアルを参照しないでも使いこなせるようになっている。また，ＡＲＴの機能ではないがZmacs にもヘルプ機能が付いている。

(6) その他

以上のようなデバッグ環境，及び随時Zmacs エディタを呼び出しインクリメンタルコンパイルできるような環境をＡＲＴではＳＴＵＤＩＯと呼んでいる。

2.5.2 ＫＥＥ

(1) エディタ

ＫＥＤＩＴというエディタを持つ。これはＫＥＥの中からZmacs エディタを制御しているものであるが，ユーザーには，ＫＥＥ専用のエディタのように見える。名前をＫＥＤＩＴという別名にしているだけの価値はある。ＫＥＤＩＴは，図2.3－1のように，ＫＥＥのＬＩＳＰ表現を直接操作するのに便利である。拡張コマンドは使用できないが，他はZmacs と同一である。メソッドやルールの編集及びユニット，スロット単位のコピーに使用した。

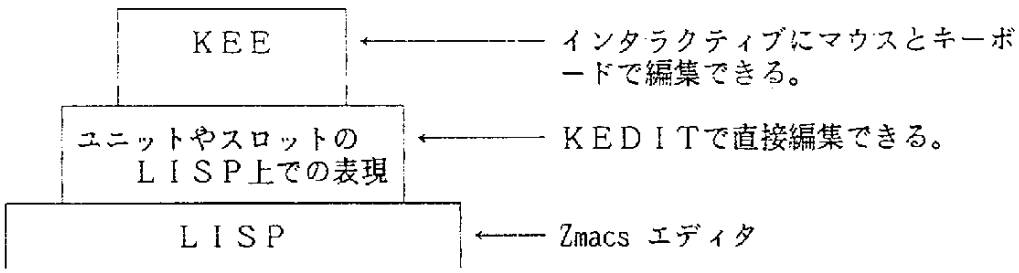


図2.5－1 ＫＥＥの知識ベース編集の環境

Zmacs エディタも，直接使用できるが，これはＫＥＥがＬＩＳＰ上でどのように表現されているかを知らないと，使用困難である。メソッドやルールの中で呼ばれるＬＩＳＰ関数の編集には使用できる。

(2) インタラクティブな編集機能

マウスを使って，スロットやユニットをポインティングすることによって，それらのスロットやユニットに対するブラウズ及び編集を，メニュー方式で行なえる。キーボードからは最低限のことを入力してやれば良い。この編集機能だけで，知識ベースのほとんどの部分を構築できる。

(3) 整合性のチェック

ＫＥＤＩＴから抜けたとき及びインタラクティブな編集の最中に，次のようなチェックが行なわれる。

① 未定義スロット，ユニットの指摘

② スロット値のデータ型（数値型，文字列型，リスト型，メンバー型など）の制約チェック

③ スロット値の個数（シングルバリュー，マルチバリュー）の制約チェック

④ スロット値の範囲の制約チェック

これらのチェック機能はフラグによって黙らせることもできる。

また，知識ベースをファイルにセーブする際，メソッドの部分は抽出されて，コンパイルされる。このコンパイルの際に次のチェックが行なわれる。ただし，このコンパイルはＫＥＥの機能ではなく，ＬＩＳＰ処理系の機能である。ＡＲＴのコンパイルがReteアルゴリズムでネットワークを作成するのを意味しているのに対し，ＫＥＥのコンパイルは通常のＬＩＳＰのコンパイルを意味している。

①′未参照変数，引数の指摘

②′未定義関数の指摘

(4) デバッグ環境

ＫＥＥの思想として，開発時はインタープリターで，運用時にコンパイルされたコードにするという考えがある。インタープリターであるから，全般的にデバッグが容易である。

プロダクションルールのデバッグに対しては，ＡＲＴと類似の機能が多様に備っている。

メソッドのデバッグに対しては，特別な機能はない。しかし，実行中にも(3)の①～④のチェックが行なわれること，及びトレースしたいスロットにデーモンをしかけておくことなどで不便は感じなかった。

(5) ヘルプ機能

ヘルプ機能はない。しかし，すべてメニュー形式であること，キーボード入力のときは適切なプロンプトが出ること，及び途中で中止（Abort）できることなどから，ほとんど不自由さは感じなかった。

2.5.3 ＫＣ

(1) エディタ

ＫＣは，独自に，スキーマ間の関係の編集のためにPalmエディタ，及びスキーマの編集のためにCoconut エディタを持つ。ＶＡＸ／ＶＭＳのＶＴ100 の環境下でも，これらのエディタの一部の機能は使えるが，もともとマウスとマルチウィンドウを前提としたエディタであるので，その機能は充分生かされなかった。

今回の環境では，ＥＤＴエディタ，ＧＥＭＡＣＳエディタ及びＶＡＸ－ＬＩＳＰの提供するエディタが使えたので，PalmやCoconut エディタは，ほとんど使わなかった。

(2) インタラクティブな編集機能

スキーマに関して，(1)で述べたPalmやCoconut エディタがある。

(3) 整合性のチェック

ＫＥＥと同様なスロット値のチェック機能がある。

(4) デバッグ環境

開発時は，インタープリターであるので，デバッグには有利である。

ＣＲＬ－ＯＰＳは，ワークベンチというデバッグ環境を持っているが，これもＶＴ100からは，充分にその機能を使えなかった。しかしそれでも一般的なＯＰＳ５と同等以上のデバッグ機能はもっている。

ＣＲＬ－ＰＲＯＬＯＧも，ワークベンチというデバッグ環境を持っているが，今回はプロローグは使用していないので使っていない。

(5) ヘルプ機能

Palm及びCoconut エディタに，ヘルプ機能がある。

2.5.4　まとめ

いずれのツールとも，優れた開発環境を提供している。

表2.5－1に結果をまとめた。

表2.5－1　ツールの開発環境の比較

項目＼ツール		ART	KEE	KC
	ツール外のエディタ（OS，第3者供給）	Zmacs エディタを，ARTの中から随時呼び出し，変更・追加した個所だけインクリメンタルコンパイルできる。	Zmacs エディタをLISP関数の編集に使える。KEEの知識ベースを編集するにはKEEの内部表現を知らないとできない。	EDT，GEMACSエディタ VAX－LISPエディタ
	ツールの提供するエディタ	な　し	KEDITというエディタがある。（注） KEEの内部表現を知らなくても，ユニット単位，スロット単位の編集がまとめてできる。	スキーマ間の関係の編集にPalmエディタがある。 スキーマの編集にCoconut エディタがある。
マウスとキーボードを使ったインタラクティブな編集機能		な　し	ある。KEEの初心者でも簡単に使える。	上記のエディタは，インタラクティブである。
整合性のチェック		コンパイル完了時（実行直前）には，スペリングミス，シンタックスエラーはほとんど完全に除ける程度のメッセージが出る。	編集及び実行時を通じて，スロット値の型，個数，範囲のチェックが行われる。ルール内，メソッド内のスペリングミス，シンタックスエラーは，実行してみないと除けない。	編集及び実行時を通じて，スロット値の型，個数，範囲のチェックが行われる。さらに，そのエラー処理の手続きをユーザーが記述できる。
デバッグ環境		非常に多様な機能をそろえており，ほとんどマウスだけで操作できる。	インタープリターであるので，有利。 メソッドのデバッグに対しては，特別な機能はないが，ルールのデバッグに対してはART同様の機能を持つ。	インタープリターであるので有利。 CRL－OPS，CRL－PROLOGそれぞれにワークベンチを持ち，一般のOPSやPrologよりもデバッグ環境はすぐれている。
ヘルプ機能		あり	なし	あり
備考		Symbolics 3600 （KEEも同様）	（注）実際には，ZmacsをKEEの中で制御しているものであるが，外見的にはあたかもKEE専用エディタのように見える。	VAX－780／VMS． 端末VT100

2.6　その他

2.6.1　ツールの頑健性

ＫＥＥ (Ver 2.0)では，「電気設備のトラブルシューティング支援問題」をインプリメントしたが，このバージョンは極めて安定した動作をしていた。

ところが，ＫＥＥ (Ver 3.0)で，「レンズの骨組み設計支援問題」をインプリメントしたところ，2.5節でも述べたように，しばしば，原因不明のシステムダウン（ツールのバグと思われる）のトラブルが発生した。たとえば，横型探索では，世界の深さが2レベルになったところで，'Ran out of swap memory'のメッセージが出てシステムダウンする。また縦型探索でも，深さが10以上になるとシステムダウンする。したがって，これらの深さに達する前に解が発見されればうまくいくが，そうでなければマシーンの再立上げが必要であった。このため，ＫＥＥ (Ver 3.0)ではデバッグ作業に相当手間取った。

また，ＡＲＴ (Ver 2.0)についても，2.5節で述べたように，(merging nil) の時，Viewpoint がある一定のレベルまで深くなるとシステムダウンする。

ＡＲＴのViewpoint 機能とＫＥＥのWorld 機能は，従来のツールにはない高度な機能を実現している点では評価できるが，現状ではまだそれらの機能はやや不安定である。また，これらの高度な機能を，一般のユーザーが正しく使いこなせるようになるには相当の経験と努力が必要と思われる。

2.6.2　開発環境

ＡＲＴとＫＥＥは Symbolics社のLispマシーン上で稼働しており，シングルユーザーで大きなメモリーを有効に使うことができる。また，ビットマップディスプレイとマウスを用いた開発環境が提供されており，ツール自体のユーザーインターフェースも良好である。

一方，ＫＣはＶＡＸ11／780 上で稼働しており，マルチユーザーのＴＳＳ環境下で使用した。Lispマシーンの開発環境とそのまま比較するのは適当ではないがツール自体のユーザーインターフェースはマウスを用いた場合と比べると，すべての操作をキーボードから行わなければならず，エディトとデバッグともに操作性が悪い。したがって，プロトタイプの作成を含めてプログラム生産性はＶＡＸ上のほうが低いと思われる。

2.6.3 スピード

ツールの推論スピードについては，Symbolics 上のツールＡＲＴ (Ver 2.0)とＫＥＥ (Ver 3.0)によって，同一問題「レンズの骨組み設計支援問題」をインプリメントしたので，おおよその比較ができる。この問題は，ルールベースのシステムとして，ＡＲＴのViewpoint 機能とＫＥＥのWorld 機能をもちいてインプリメントされており，ルールとのマッチングのスピードを比較するには適している。

結論はＡＲＴの方がＫＥＥよりも，はるかに速い。ＫＥＥが遅い理由はReteアルゴリズム使っていないこと，及びインタープリターであるためと思われる。

ＫＣでも同じ問題をインプリメントしたが，ＫＣがＶＡＸ上で稼動しており，ハードウェアがそのものがＡＲＴやＫＥＥと異なるために，他のツールとスピードを比較することはあまり意味がない。

付録 1.　エキスパートシステム開発ツール評価標準問題の概要

（付録1）エキスパートシステム開発ツール評価標準問題の概要

1. レンズの骨組み設計支援問題

1.1 レンズの設計工程

レンズ設計には以下の5つの工程がある。

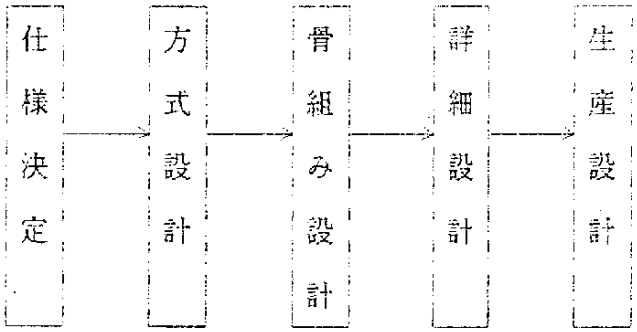


図1.1－1　レンズ設計の工程

仕様決定　：　市場性などを考慮して，機能（焦点距離，ズーム比，フィルムサイズ），性能（各種収差，解像能），形状寸法（全長，重量）などを決定する。

方式設定　：　機能，性能の要求仕様を満たす方式の選定をおこなう。たとえば，各種ズームタイプの選択など。

骨組み設計　選定された方式に対する骨組みを，主として機能，形状寸法から決定する。たとえば，レンズ群の焦点距離や群間の間隔など。

詳細設計　：　骨組みへ実際のレンズ形状を与え，各種収差を許容量以下におさえる。

生産設計　：　製造上の各種標準へ適合させるための設計。

本設計問題では，骨組み設計の過程を支援するエキスパートシステムの作成を考える。その為，方式設計に関する選定はすでに完了しているものとして機能仕様が与えられる。次章の骨組み設計では，機能仕様で仮定する1つのズームタイプ（キャノンタイプ）を中心に説明をこおなう。

1.2　レンズの骨組み設計

1.2.1　骨組みの役割

複数のレンズは，等価変換により１枚のレンズとみなすことができる。そういった複数のレンズの集りをレンズ群と呼ぶ。レンズ群は，レンズの組合せにより凸（とつ）群と凹（おう）群の２種類に分類できる。

ズーム・レンズは以下の図のように，違った機能を有する複数のレンズ群により形成される。図1.2 - 1は５つのレンズ群より成るズームレンズの機能図である。

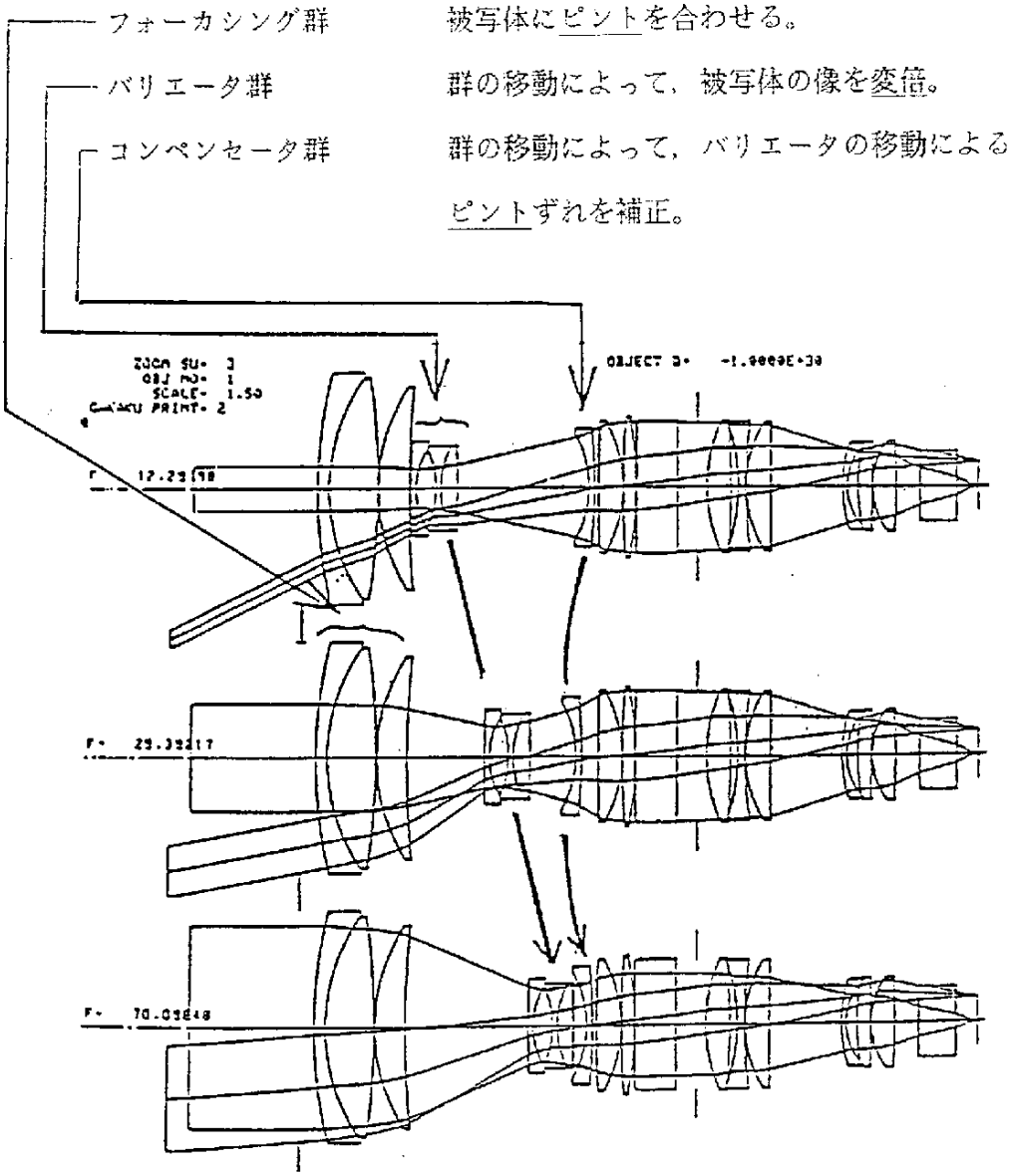
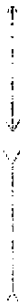


図1.2－1　ズーム・レンズの機能図

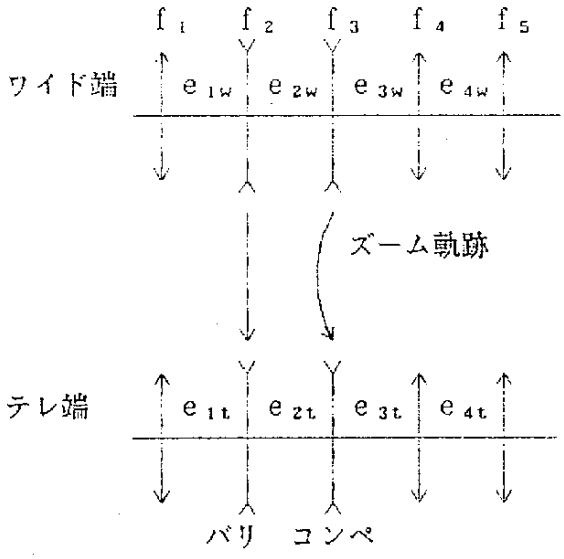
ズームレンズの各レンズ群を1枚のレンズへ等価変換し，これに以下のシンボルを与える。

とつ群：

おう群：



本設計問題で扱う，ズームレンズは5つのレンズ群をもっており，図1.2－2に示すパラメータによって骨組みを表現することが出来る。このパラメータを骨組みパラメータと呼び，ズームレンズの基本的な機能，形状を規定する。



ただし，

f_1 , f_2 , f_3 , f_4 , f_5 は各群の焦点距離

e_{1W}, e_{2W}, e_{3W}, e_{4W} はワイド端での各群間隔

e_{1t}, e_{2t}, e_{3t}, e_{4t} はテレ端での各群間隔

更に

ズーム比　$Z = \dfrac{f_t}{f_w}$

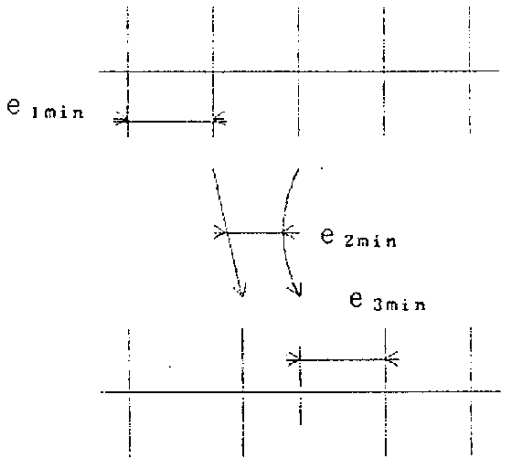
図1.2－2　ズーム・レンズの骨組みの表現

1.2.2　骨組み設計の目的

骨組設計の主要な目的は，与えられたズームタイプの枠の中で，以後の詳細設計にて形状設定・各収差補正等の実現が可能であり，かつ骨組としてより軽量・コンパクト・低コストなものを発見することである。

(1)　形状設定・各収差補正等の実現上の制約

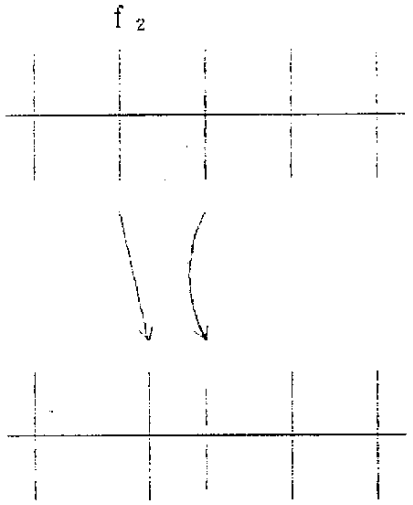
形状設定上の制約の例



ズーミングに際し，第1～3群の後の間隔の最小値（e_{1min}, e_{2min}, e_{3min}）が，ある一定値より小さくなってはいけない。

(→実際のレンズ群を設定するスペースを確保するための，経験的な限界値の存在)

収差補正上の制約の例



第2群の焦点距離 f_2 が，きつすぎると収差補正上困難となるため，経験的限界値が知られている。

その他の制約の例として，

光線の条件　たとえばレンズの明るさ f_{no} の限界値

——→　実用的なレンズの明るさを確保するため。

(2) 目標条件

骨組として，より軽量・コンパクト・低コストなものの評価基準としては，経験的に以下の事実が知られている。したがって，今回は以下の2点を目標条件とする。

〔目標条件1〕

<u>第1群の有効径EA_1が一定置より小さなもの。</u>

有効径とは，各群を通る光線の中で，ズーム範囲中最外周を通る光線の作る円の口径。(図1.2－3参照)

第1群はレンズ内で最も体積が大きいため，ガラスの重量比，材料比に大きく影響する(注1)ことが経験的に知られている。

第1群の有効径EA_1 ⟶ ガラス体積減少 ⟶ 軽量比
ガラス体積減少 ⟶ 低コスト化

を限界まで小さくする。

〔目標条件2〕

<u>全長Lが一定置より小さいもの。</u>

全長Lとは各群間との間隔（e_1，e_2，e_3，e_4）及び，最終群からフィルム面までの間隔（バックフォーカス）の和で決まる。(図1.2－3参照)

全長Lを限界まで小さくする。 ⟶ コンパクト化

(注1) 第1群のガラス重量は，全ガラス重量の50～70%を占めているといわれている。

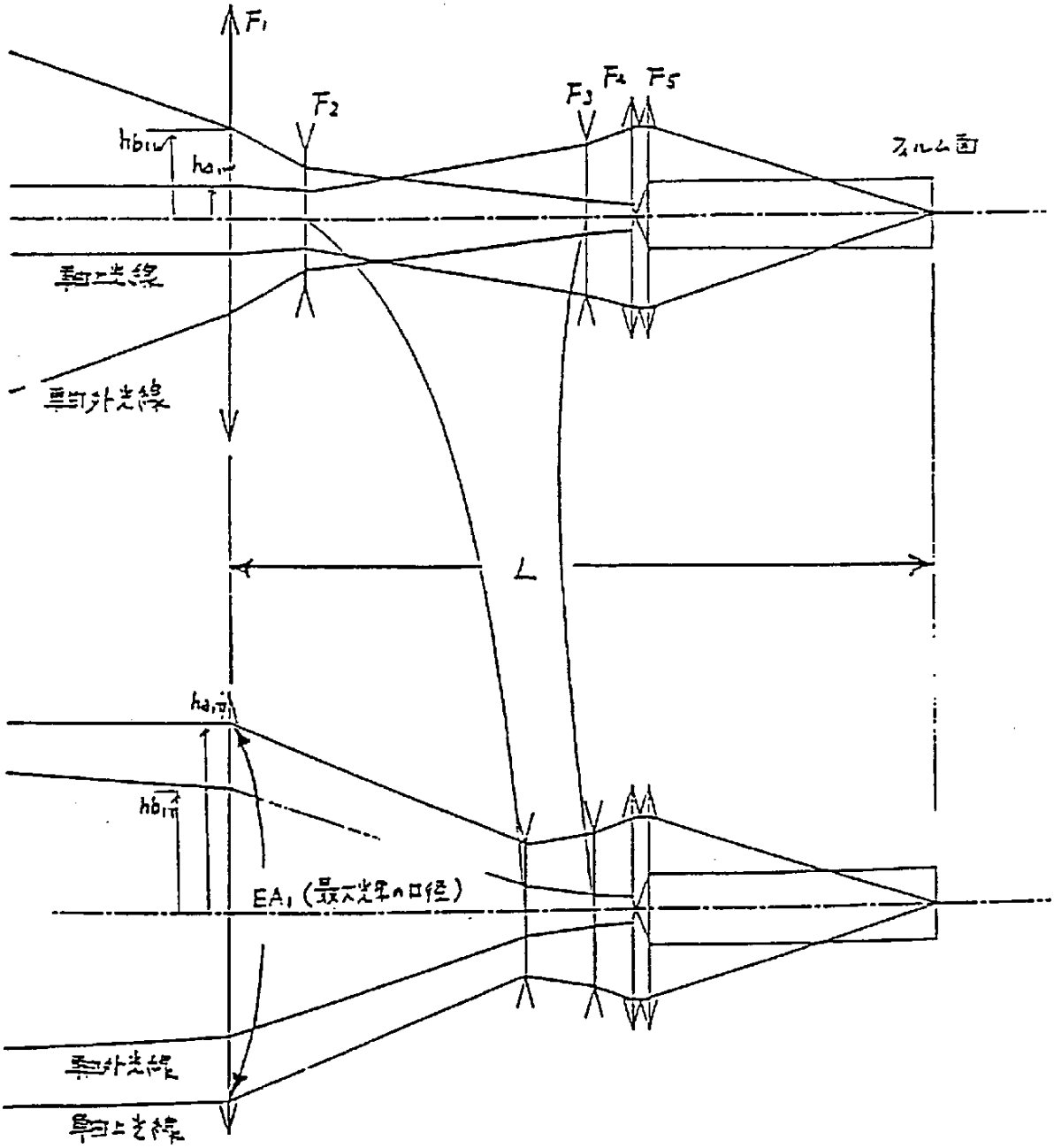


図1.2－3　有効ＥＡ1と全長Ｌ

1.2.3　骨組み設計の過程

骨組み設計は以下の４つの過程に分けて考える事ができる。

① 詳 細 化　：　仮説から実際の設計対象を構築する知識により骨組を作り出す。

② 評　　価　：　骨組の性能を知るため，シミュレーション手順やパラメータの与え方の知識により，性能等の評価結果をうる。

③ 原因分析　：　上記評価結果から，目標条件を満足しない原因を分析する知識により，定性的に原因を分析する。

④ 仮説更新　：　上記の原因を取除くため，仮説内容を修正する方法の知識により，仮説を更新してトライ＆エラーをくり返す。

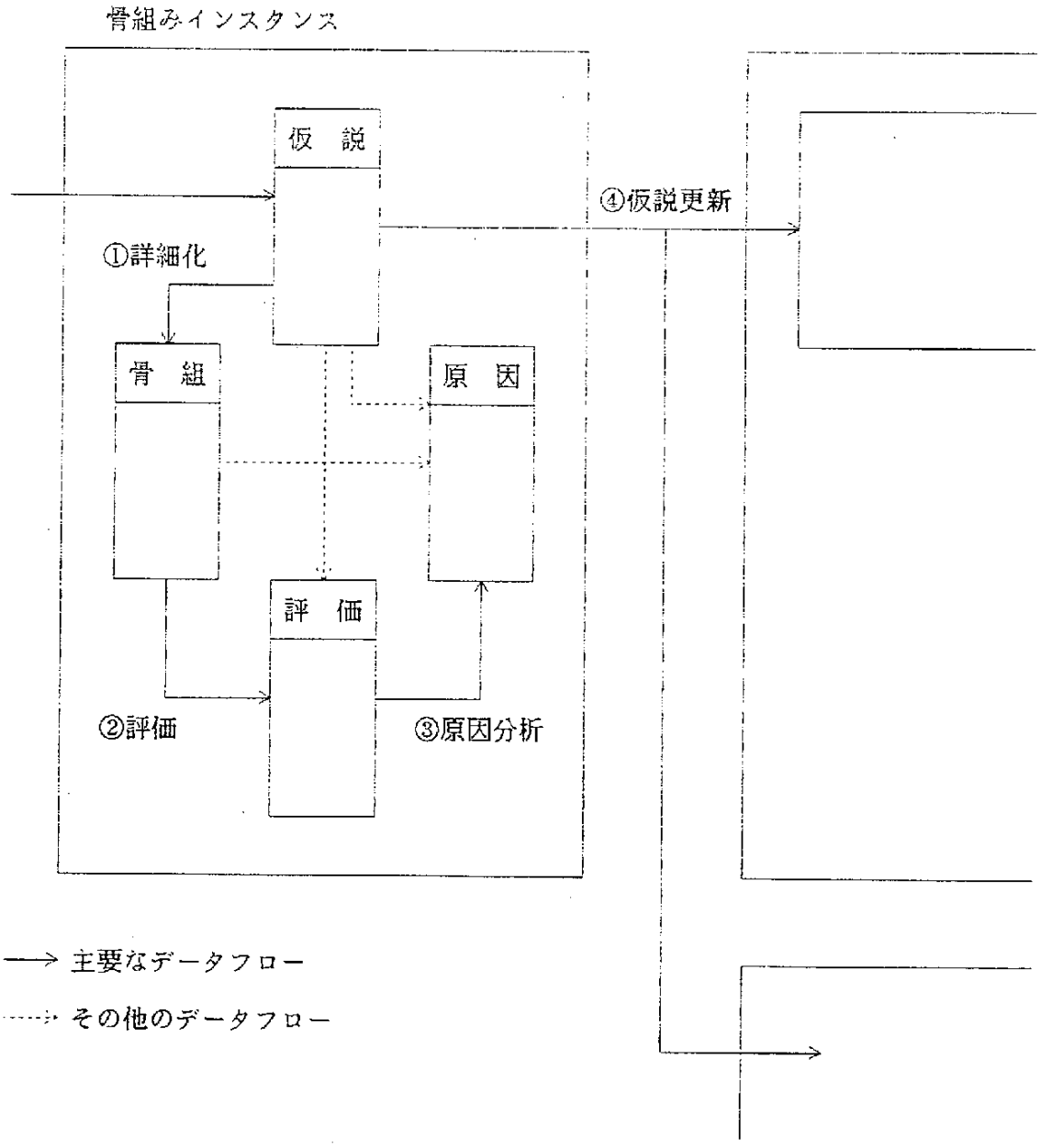


図1.2－4　設計過程の概略

骨組み設計問題は，ある制約条件下で最適評価を考えるパラメータ群をみいだす一種の非線形計画問題ととらえることができる。しかしながら，シミュレーションを用いるために各パラメータ間の関係を与える関数を決定するのが困難であるとともに，求められたにしてもその非線形度が極めて高いことが予想されるなど，数値計算による従来の非線形計画問題に帰着できず，レンズ設計専門家による経験によってパラメータの修正が行なわれている。本システムでは，専門家によるパラメータ修正の経験をルール化し，制約条件を充す骨組みパラメータを図1.2－4のサイクルの中で設定していく。

1.3　設計支援システムとしての機能

1.3.1　基本機能

本システムの基本機能は，ある与えられた初期条件（仮説パラメータ）から制約条件を満足する骨組みパラメータ，すなわち5つのレンズ群の焦点（f i）とテレおよびワイド時における各レンズ群間の間隔（$e_{i,t}$，$e_{i,w}$）を決定することが目的である。

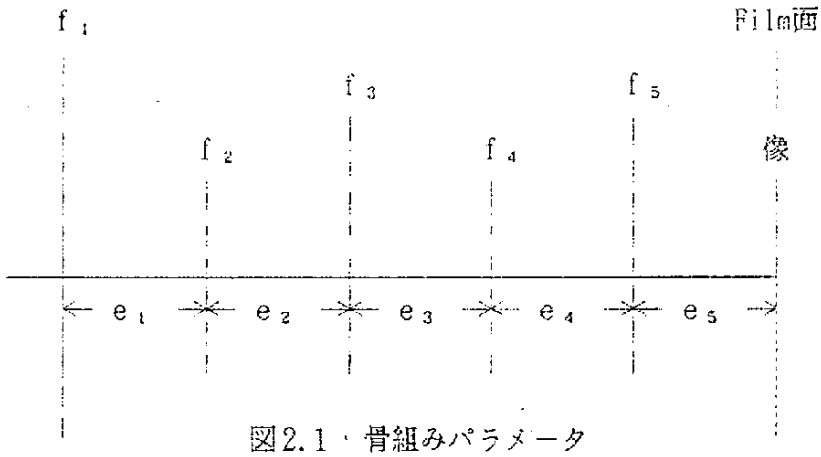


図2.1　骨組みパラメータ

解は複数存在することもありうる。

1.3.2　付加機能

基本機能では，単に与えられた初期条件と制約条件から，それを満す骨組みパラメータを求めることを要求している。付加機能としては，実際にレンズ設計者が本システムを利用するにあたって，あれば望ましいと思われる機能について述べている。よって付加機能はかならずしも実現しなければならないといったものではなく，本システムのインプリメントに使用するツールのレベルに応じて実現のレベルは異なると考えられる。

1.3.2.1　各種パラメータの表示機能と変更機能

レンズ設計は基本的には各種のパラメータの決定過程であり，方式設計→骨組み設計→詳細設計という段階を経て決定されていく。このようなパラメータの決定過程はかならずしも決定的に進むものではなく，設計者は各段階をフレキシブルに行き来しながら設計を進めていく。従って，各段階でのパラメータをユニークに固定しながら決定していくのでなく，かく段階での決定には複数の解があるのが当然である。解を選択的に表示したり，初期条件（すなわち方式設定から渡されるパラメータ）や制約条件を与えるパラメータを変更できる機能も必要である。

(1) 初期条件（仮説パラメータの初期値）の表示と変更

解探索の初期値を与える仮説パラメータの初期値は，あらかじめデフォルト値が設定されている。システムで与えられたデフォルト値の表示，並びにユーザからの要求に応じての変更機能を提供する。ユーザが変更するのは各ＲＵＮのスタートの時点においてである。

(2) 制約条件（制約値）表示と変更

解探索の制約条件を与えるパラメータの制約値はあらかじめデフォルト値がシステムで設定されている。制約値のデフォルト値の表示とユーザからの要求に応じてその変更機能を提供する。表示と変更はシステムのＲＵＮ中に適宜おこなえるものとする。

(3) 探索条件パラメータの表示と変更

仮説パラメータを修正しながら解の探索をおこなう際に，探索をどちらの方向にどの位のステップ量で進めるかを与える探索条件パラメータの表示とシステムであらかじめ与えられているデフォルト値の変更をおこなう。表示と変更はシステムのＲＵＮ中に適宜おこなえるものとする。

(4) 各種パラメータの表示

設計過程の進行状況を適宜ユーザに表示する。表示すべきパラメータとしては以下のようなものがある。

・仮説パラメータ

・骨組みパラメータ

・評価対象パラメータ

・原因パラメータ

表示は仮説（骨組みインスタンス）毎にユーザの要求に応じておこなう。

1.3.2.2　解探索の制御

1つの仮説（骨組みインスタンス）から更新される仮説は通常複数存在する。どちらの方向に更新を進めてゆけば解により速く到達できるかというのは明確にはわからない。このような状況下での解探索の支援機能を考える。

(1)　ユーザとのインタアクションによる制御

複数の仮説更新候補が出た時点でどの候補からさらに評価を進めてゆくかをユーザにインタアクティブに選択させる。各種パラメータの表示機能と変更機能を組合せることにより解探索の支援をおこなう。

(2)　評価関数の使用

仮説を生成しながら可能解を求めるという問題をより効率的に解くためにヒューリスティクな評価関数の使用を考える。評価関数を使用することによって，

・探索の刈り込み

評価関数値がある一定値より小さい仮説については刈り込みを行なう。

・探索順序の制御

評価関数値の高い仮説から探索を進めることにより，より効率的に最良解から順に解を求めることができる。

といったことがおこなえる。しかし，現在の時点では評価関数に関する仕様は与えられていない。

1.3.2.3　説明機能

(1)　トレース機能

ある解が得られたとき，その解がどのような過程を経て得られたのか，仮説の変化の記述としてユーザに表示する。仮説の変化の記述としては，

・使用した原因分析ルート

・使用した仮説更新ルール

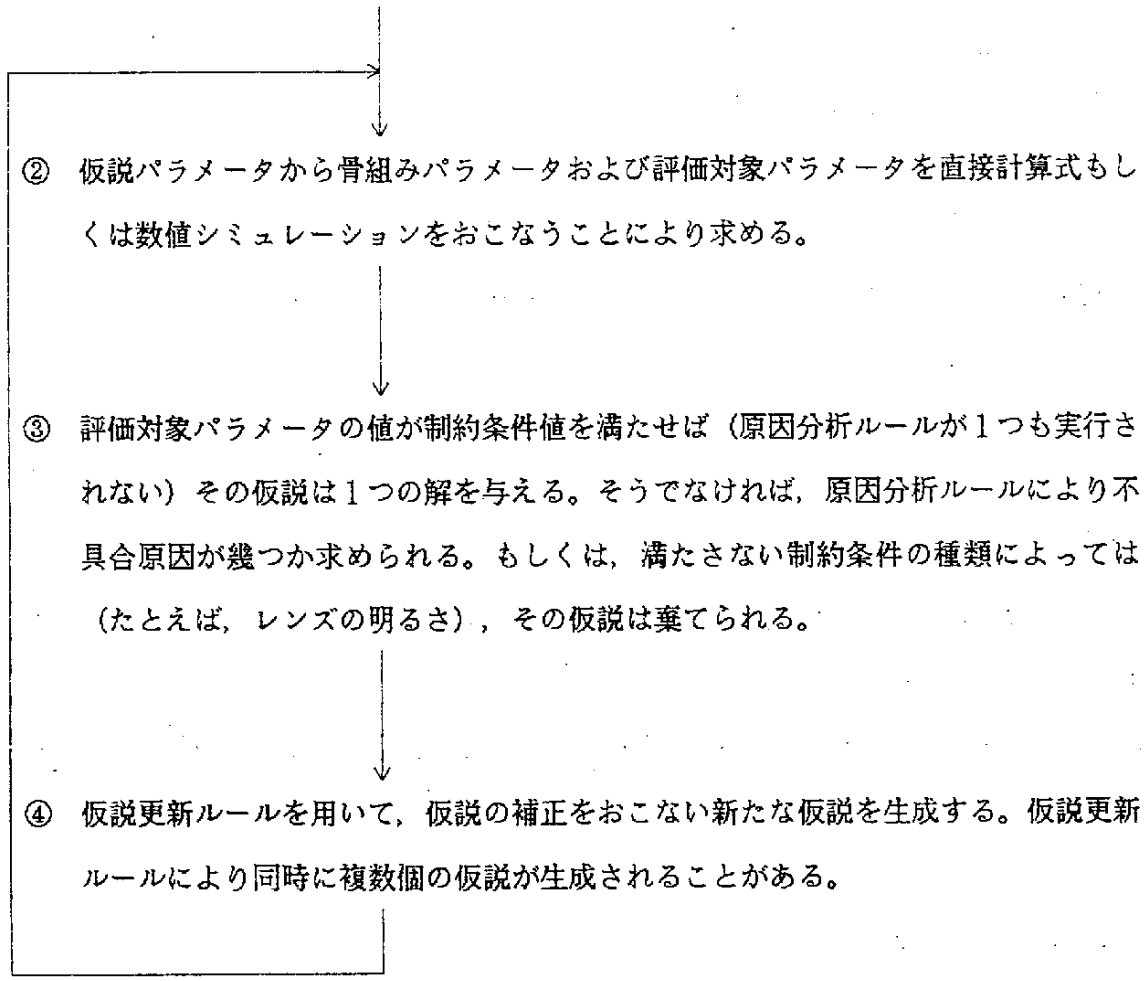
なども含めて表示する。

1.4　処理概要

1.4.1　処理の流れ

骨組み設計の過程（1.2.3）に従って以下のような処理の流れとなる。②，③，④は解が得られるまでサイクリックに繰り返される。

① 仮説の初期パラメータを決定する。通常は仮説パラメータとしてシステムの与えるデフォルトの初期値を仮定し，最初の仮説とする。



1.4.2　各処理の概略

(1)　初期パラメータの決定

初期の仮説パラメータとして，本問題ではデフォルトの値を与える。本来は骨組み設計以前の段階での要求仕様を満たすべく与えられるものである。仮説パラメータとしては，ズーム比，ワイド端での全系焦点距離，バリエータ焦点距離，一部骨組みパラメータの典型的な値などが含まれている。また，骨組み評価用の評価対象パラメータの制約条件値と仮説更新時に仕様する探索条件パラメータもデフォルト値が与えられる。

(2) 骨組みパラメータと評価対象パラメータの計算

骨組みパラメータは，各レンズ群の焦点距離と，ワイド端およびテレ端におけるレンズ群間の間隔より成る。これらは，仮説パラメータから数値計算により求められる。評価対象パラメータとしては，全長，第1群有効径の最大，レンズの敏感度，レンズ群間隔の最小値などがある。これらの評価対象パラメータを求める為には，数値計算とズーム軌跡や光線追跡などのシミュレーションをおこなう必要がある。

なお，数値計算式やシミュレーション式については別途Lispで記述したものが用意されている。

(3) 不具合原因の分析

評価対象パラメータの値が制約条件値を満たさなければ，何が原因なのかを分析する。満たされないパラメータと原因を結ぶいくつかの原因分析ルールが与えられる。たとえば，全長が制約条件値以上で，1－2群間の最小間隔が制約条件値以下のとき原因としては，バリエータ（第2群）の移動量過大が考えられる，といったルールである。同じ制約条件値が満たされない場合でも，複数の原因が考えられる。

(4) 仮説の更新

不具合原因と仮説パラメータの補正方法を結ぶいくつかの仮説更新ルールが与えられる。たとえば，バリエータの移動量が過大の場合は，バリエータの焦点距離（f_2）を少し（探索条件パラメータで与えられる）だけ短かくしてみる，といったルールである。同じ不具合原因から複数の補正方法が導かれる。よって更新はツリー状に広がってゆくことになる。

2. 電気設備のトラブルシューティング支援問題

2.1 問題の概要

2.1.1 分 野

設備診断

2.1.2 対象設備

サイリスタレオナード装置

2.1.3 問 題

保護装置が作動して電動機が停止して，ＳＣＲゲート遮断が発生した場合を考える。

2.1.4 目 的

サイリスタレオナード装置のトラブルシューティング作業（情報収集，計測器接続，原因追求，復旧，試運転）をＣＲＴとの対話形式でオフラインにて支援する。

2.1.5　処理の流れ

処理の流れの概要を図2.1－1に示す。

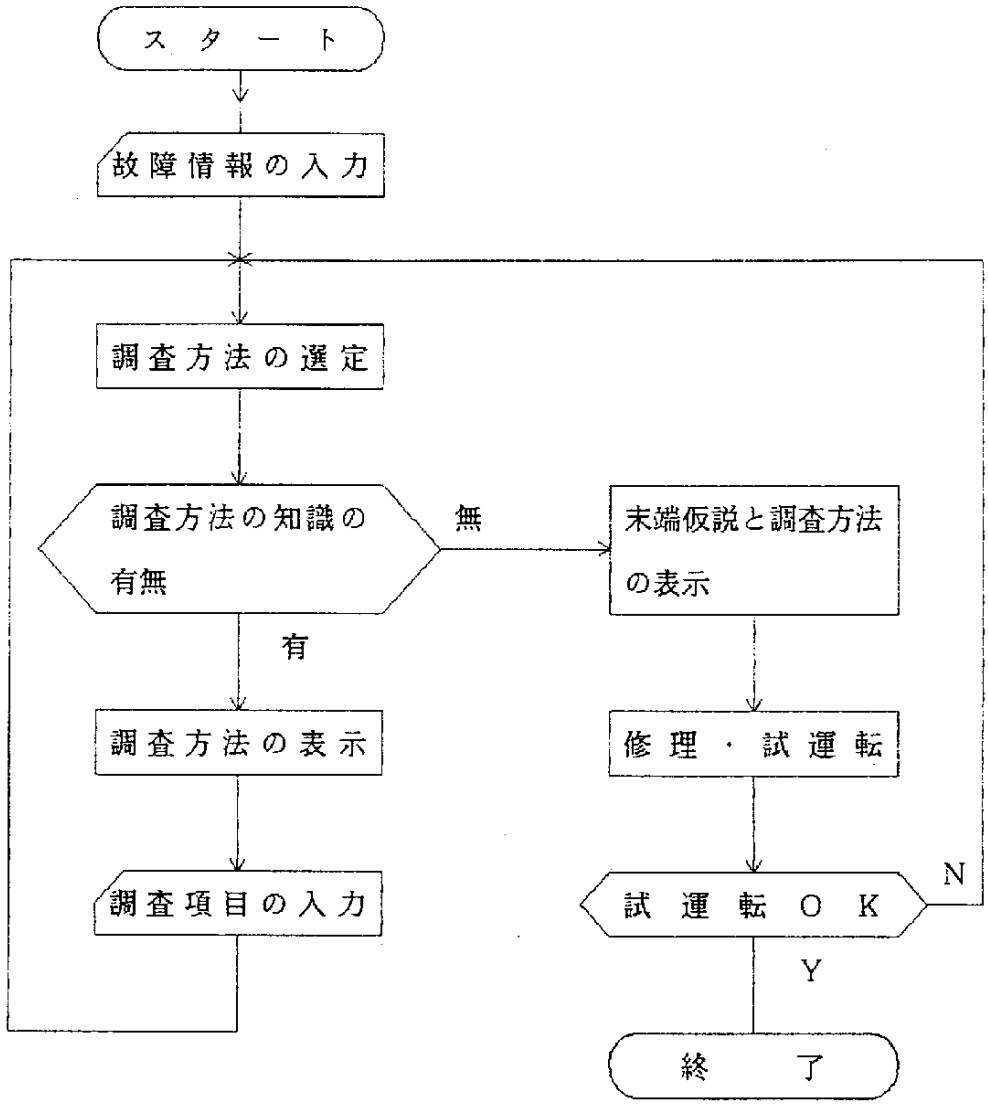


図2.1－1

2.1.6　対話イメージ概要

次にシステムの実行時のエンドユーザーとの対話イメージを示す。処理の流れと対比することによりシステムの動作の概要が理解されよう。

診断開始

ＳＣＲゲート遮断の原因を調査します。

故障情報入力

パネルを見て，次の故障情報を入力して下さい。

1．ＳＣＲヒューズ１個溶断　　〔y／n〕〔y〕

2．ＳＣＲヒューズ２個溶断　　〔y／n〕〔n〕

・・・

8．ゲート遮断表示のみ　　〔y／n〕〔y〕

仮説の評点を表示

ＳＣＲゲート遮断の中間仮説と評点は次の通りです。

仮説	評点
1．交流側の過電流・ＳＣＲ素子不良	80
2．交流側の過電流・配線短絡	50
・・・	
8．界磁回路異常	0

調査方法の選択

調査方法を選択して下さい。

調査方法〔仮説・評点〕	優先度
1．ＳＣＲ素子の絶縁抵抗測定 〔交流側の過電流・ＳＣＲ素子不良　80〕	33
2．アナログ回路の調査 〔交流側の過電流・ＳＣＲ素子不良　80〕	18
3．電源電圧の測定 〔交流側の過電流・ゲートパルス不良　3〕	10

・

・

・

番号を入力〔3〕

調査の実行

調査方法：　電源電圧の測定

電源電圧を測定して，値を入力して下さい。

電源電圧＝〔 112〕

他の調査

試運転の結果を入力

調査方法：　ＳＣＲ素子交換

部品交換後，試運転をして，その結果を入力して下さい。

試運転良好　〔ｙ／ｎ〕　〔ｙ〕

診断終了

試運転の結果が良好なので診断を終了します。

2.2 エキスパートの知識

2.2.1 フォールトツリー

故障原因仮説及び，それを検証するための調査方法は，ツリー構造として階層的にまとめられている。故障に関するこの種のツリーを以下フォールトツリーと呼ぶ。フォールトツリー上の各ノードには，仮説や調査方法が対応させられている。

調査方法に対応するノードは，調査の実行のしかたにより，テーブルタイプ，フロータイプ，因果タイプ，リーフタイプの４つに分類される。これらのノードタイプの定義は次の通りである。

(1) テーブルタイプ

ユーザーが調査方法を自分で選択できる場合。

(例)

ＳＣＲゲート遮断の調査

波形状況によるゲート
パルス発生回路の調査

波形状況によるゲート
パルス発生回路（ＶＣ以外）
の調査

ＳＣＲ素子の絶縁抵抗測定
とカーブトレーサー

(2) フロータイプ

あらかじめ定められた手続きを実行して，その結果にもとづいて次の調査がシステムによって自動的に決定される場合。

(例)

位相制御回路の調査

アナログ回路の調査

ＳＣＲ素子の調査１,2,3

(3) 因果タイプ

狭い範囲の故障調査を行う場合で，ＡＮＤ／ＯＲツリーで表現されたツリー上を順次調べる場合。この種のＡＮＤ／ＯＲツリーを以下因果ツリーと呼ぶ。

(例)

自動制御（ＶＣ以前）の調査

(4) リーフタイプ

末端調査に対応して，これより下に仮説と調査方法がない場合。

(1)，(2)，(3)以外の場合である。

（例）

電源電圧の測定

ケーブル電線の点検，絶縁測定

ミルモーターの点検と絶縁測定

リレー盤のリレー動作確認

など(1)，(2)，(3)の例以外の調査。

2.2.2　フォールトツリーの構造

(1)　ＳＣＲゲート遮断の調査

図2.2－1にルートノード「ＳＣＲゲート遮断の調査」のツリー構造を示す。

この調査は，テーブルタイプなので，仮説に対応する調査＊Ｂ１，..，＊Ｂ８のどれかをユーザーが自主的に選択しなければならない。ただし，＊Ｂ２，＊Ｂ３，＊Ｂ４の調査は子ノードをもち，それ以外の＊Ｂ１，＊Ｂ５，＊Ｂ６，＊Ｂ７，＊Ｂ８はリーフタイプノードである。

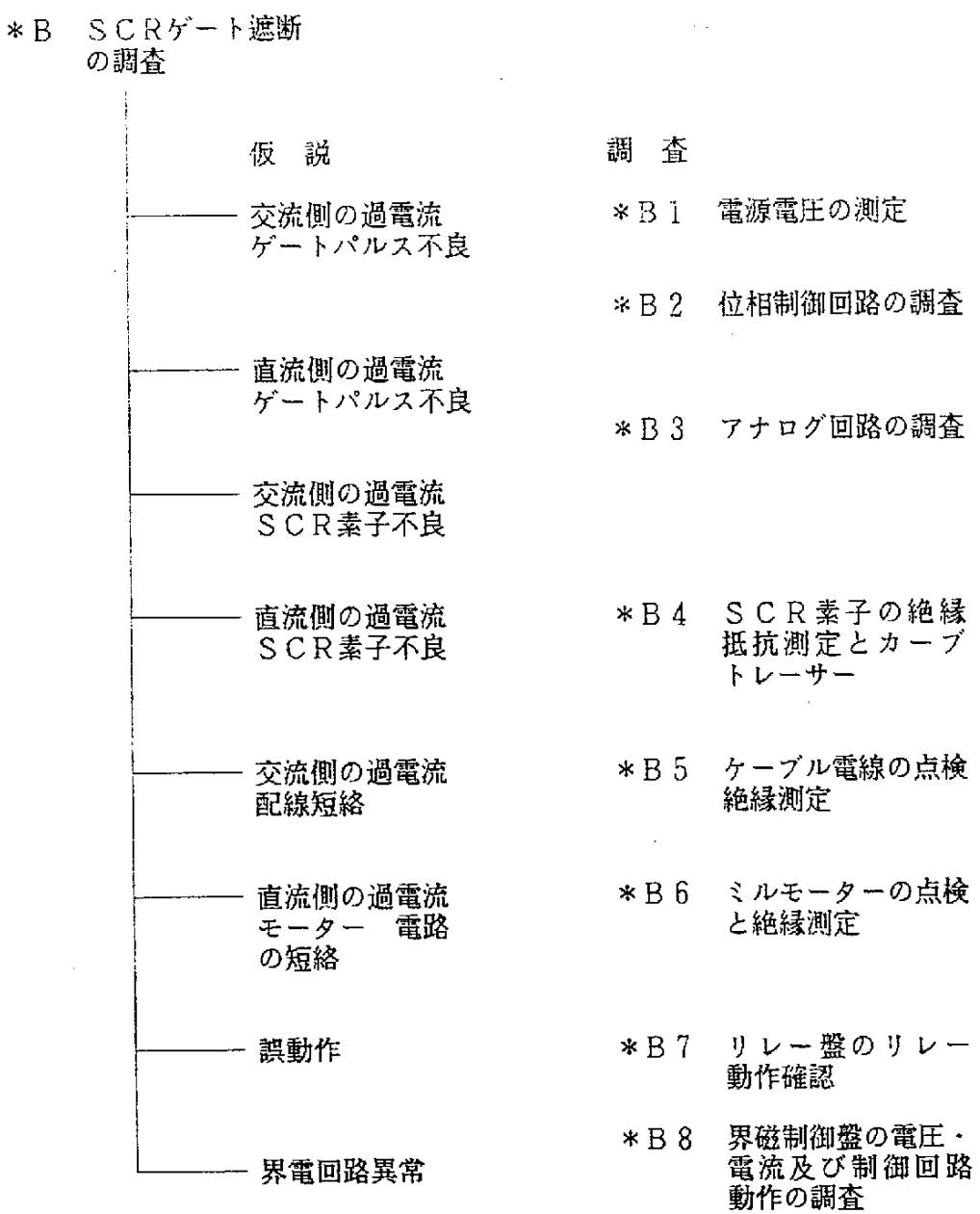


図2.2－1

(2) 位相制御回路の調査

図2.2－2にフロータイプの例として，位相制御回路の調査を示す。この場合は，ある手続きを実行して，その結果にしたがって，＊B310 のテーブルタイプの調査か，＊B21の因果タイプの調査へ分岐する。ただし，＊B2自身の調査の結果が正常ならば，下へ分岐せずに，＊B1，．．．．，＊B8のなかの他の調査をユーザーが自主的に選択実行することになる。一般に，他のフーロタイプのときも同様である。

＊B2　位相制御回路の調査
- ＊B310　波形状況によるゲートパルス発生回路（ＶＣ以外）の調査（＊B31からＶＣカードを除く）
- ＊B21　自動制御（ＶＣ以前）の調査

図2.2－2

(3) アナログ回路の調査

図2.2－3にフロータイプの例とて，アナログ回路の調査を示す。

＊B3　アナログ回路の調査

フロー

＊B31　波形状況によるゲートパルス発生回路の調査

仮　説	調　査	
ＯＴ－Ｔｒ不良	＊B311	ＯＴ－Ｔｒの調査
ＧＴ－Ｔｒ不良	＊B312	ＧＴ－Ｔｒの調査
同期電源回路不良	＊B313	同期電源回路の調査
ＶＣカード不良	＊B314	ＶＳカードの調査
ＧＳＰカード不良	＊B315	ＧＳＰカードの調査
ＧＰＧカード不良	＊A471	ＧＰＧカードの調査
Pulse-Ｔｒ不良	＊B317	Pulse-Ｔｒの調査

図2.2－3

(4) ＳＣＲ素子の絶縁抵抗測定とカーブトレーサー

図2.2－3にテーブルタイプの例として，ＳＣＲ素子の絶縁抵抗測定とカーブトレーサーの調査を示す。

＊Ｂ４　ＳＣＲ素子の絶縁抵抗
測定とカーブトレーサー

仮　説	調　査	
ＳＣＲ素子全部が疑わしい	＊Ｂ42	ＳＣＲ素子の調査２
一部のＳＣＲが疑わしい	＊Ｂ41	ＳＣＲ素子の調査１
ヒューズが疑わしい	＊Ｂ43	ＳＣＲ素子の調査３

図2.2－4

2.2.4　因果ツリー

被疑範囲がある程度しぼられてきたら，このツリー上をルートノードから順次テストする。このツリーはＡＮＤノードか，ＯＲノードか，Leafノードから構成されているＡＮＤ／ＯＲツリーの一種であるが，場合によっては，全パステスト（全ノードチェック）を行う場合も有り得る。

図2.2－6に因果ツリーの構造を示す。

図2.2－6

2.2.3　フロータイプの調査

フロータイプの調査の一例として，「位相制御回路の調査」の具体的な手続きを図2.2－5に示す。この手続きの実行結果によって次に行う調査が決まることになる。

図2.2－5

2.2.5　仮説と調査項目

仮説とそれを検証するための調査項目との関連の強さ（相関関係）を定量的に表した表があり，以下それを「仮説と調査項目の表」と呼ぶ。この表は，専門家の経験的に重要な知識を抽出して得られたものである。

関連の強さは，－1.0から1.0の範囲の実数で表す。この値が1に近いほど仮説とその調査項目とは関連が強く，逆に，－1に近いほど関連が弱いことを表している。この表はユーザーがキーボードから入力した故障情報（または，調査結果）を表現するための表（以下，「調査結果表」と呼ぶ）と組み合わせて，仮説の評点を計算するために使われる。計算方法については別項を参照。

テーブルタイプの各ノードに対応して「仮説と調査項目の表」が用意されている。

表2.2－1に「ＳＣＲゲート遮断の調査」に対応する表の一部分を示す。

表2.2－1

仮　　説	交流側の過電流		
調査項目	ゲートパルス不良	ＳＣＲ素子不良	配線短絡
ＳＣＲヒューズ1個溶断	0.5	0.95	0
ＳＣＲヒューズ2個溶断	0.95	0.5	0
モーター接地	-0.5	0	0
モーター過負荷	0	0	0
界磁喪失	-0.95	-0.95	-0.95
界磁ＳＣＲヒューズ断	-0.95	-0.95	-0.95
５１動作	0.95	0.95	0.5
過速度	-0.5	-0.5	-0.95
ゲート遮断表示のみ	0.5	0.5	0.25

2.2.6　調査の容易性

各仮説の調査方法ごとに，調査の容易性，重要性等を定量的に表した表があり，以下これを「調査の容易性の表」と呼ぶ。この表は，専門家の経験的な知識を中心にして作成されたものである。

容易性等の値は1から5までの整数で表す。この値が1に近いほどその項目の優先度が高く，5に近いほど優先度が低いことを意味する。

テーブルタイプの各ノードに対応して「調査の容易性の表」が用意されている。

この表は，「仮説と調査項目の表」と組み合わせて，調査方法の優先度を計算するために使われる。計算方法は別項を参照。

表2.2－2に「ゲートパルス不良」に対応する表の一部を示す。

表2.2－2

仮説	ゲートパルス不良		
調査方法 / 評価項目	アナログ回路の調査	位相制御回路の調査	電源電圧の調査
調査の容易性	4	4	2
調査に要するコスト	4	3	2
保守体制予備品量入手の容易	3	3	2
繰り返し故障発生頻度	4	4	3
評価点数	3.7	3.4	2.2

2.2.7　仮説の評点の求め方

仮説と調査項目の表2.2－3と調査結果の表2.2－4から仮説Ｘｊの評点Ｐ（ｊ)(%)を求めるには次の式に使う。

$$P(j) = 50\,(Plus(j) + Minus(j) + 1.0)$$

ただし，

$$Plus(j) = \max\{R(i,j)\} \cdot Pro1(j) / Sum1(j)$$

$$Minus(j) = \min\{R(i,j)\} \cdot Pro2(j) / Sum2(j)$$

$$Pro1(j) = \Sigma\,|Vi \cdot R(i,j)| \qquad R(i,j) > 0 \text{についての和}$$

$$Sm1(j) = \Sigma\,|R(i,j)|$$

$$Pro2(j) = \Sigma\,|Vi \cdot R(i,j)| \qquad R(i,j) < 0 \text{についての和}$$

$$Sum2(j) = \Sigma\,|R(i,j)|$$

である。

表2.2－3

仮説 ＼ 調査項目	Ｘ1	...	Ｘｊ	...	Ｘｎ
Ａ1	R(1,1)	...	R(1,j)	...	R(1,n)
Ａ2	R(2,1)	...	R(2,j)	...	R(2,n)
...		...			
Ａｍ	R(m,1)	...	R(m,j)	...	R(m,n)

表2.2－4

調査項目	値
Ａ1	Ｖ1
Ａ2	Ｖ2
...	...
Ａｍ	Ｖｍ

表2.2－5と表2.2－6から，仮説Ｘの評点Ｐを求めるには，

$$P = 50\,(Plus + Minus + 1.0)$$

$$Plus = \frac{0.95\,(0.5 \times 1 + 0.95 \times 0 + 0.95 \times 0 + 0.5 \times 1)}{(0.5 + 0.95 + 0.95 + 0.5)}$$

$$= 0.32$$

$$\text{Minus} = \frac{-0.95\ (|-0.5\times0|+|-0.95\times0|+|-0.95\times1|-0.5\times1|)}{(|-0.5|+|-0.95|+|-0.95|+|-0.5|)}$$

$$= -0.48$$

したがって,

$$P = 50\times\ (0.32-0.48+1)$$

$$= 42$$

表2.2－5

仮説 / 調査項目	X
ＳＣＲヒューズ１個溶断	0.5
ＳＣＲヒューズ２個溶断	0.95
モーター接地	-0.5
モーター過負荷	0
界磁喪失	-0.95
界磁ＳＣＲヒューズ断	-0.95
５１動作	0.95
過速度	-0.5
ゲート遮断表示	0.5

表2.2－6

調査項目	値
ＳＣＲヒューズ１個溶断	1
ＳＣＲヒューズ２個溶断	0
モーター接地	0
モーター過負荷	1
界磁喪失	0
界磁ＳＣＲヒューズ断	1
５１動作	0
過速度	1
ゲート遮断表示	1

1.2.2.8　調査方法の評点の求め方

調査の容易性の表2.2－7から調査方法Ｙｋの評点Ｓ（ｋ）をもとめるには次の式を使う。

$$S(k) = \max \sqrt[m]{\prod_i L(i,k)}$$

ただし，Ｌ（ｉ，ｋ）の中に，１が２つ以上あれば，

$$S(k) = 1$$

とする。また，同一の調査方法があれば，max をとる。

表2.2－7

調査方法 / 調査項目	Ｙ１ .... Ｙｊ ... Ｙｎ
Ｂ１	Ｌ(1,1) ... Ｌ(1,j) ... Ｌ(1,n)
Ｂ２	Ｌ(2,1) ... Ｌ(2,j) ... Ｌ(2,n)
...	...
Ｂｍ	Ｌ(m,1) ... Ｌ(m,j) ... Ｌ(m,n)
評　点	

1.2.2－9　調査方法に対する優先度の求め方

仮説Ｘｊに対する調査方法Ｙｋの優先度Ｍ（ｊ，ｋ）は，次の式で計算する。

Ｍ（ｊ，ｋ）＝Ｐ（ｊ）／Ｓ（ｋ）

ただし，

Ｆ（ｊ）＝Ｘｊの評点，　Ｓ（ｋ）＝Ｙｋの評点

である。

2.3 エキスパートシステムの機能

2.3.1 概 要

この問題は，故障原因に対する仮説と，それを検証するための調査方法が，あらかじめ明確に与えられており，さらにそれらが階層構造を持ったツリー構造として表現されている。従って，1つの定式化としては，ルールベースのエキスパートシステムではなく，ツリー上の探索問題とみなしてシステムを構築することができる。以下，フォールトツリーの探索問題として定式化したときの，必要な機能について述べる。ただし，対話イメージを実現するための必要最低限の入出力機能については省略する。

2.3.2 基本機能

(1) フォールトツリーを探索する機能

フォールトツリーをルートノードから出発して，フォールトツリーを降りたり，昇たりする機能である。

(2) 各ノードの調査実行機能

調査方法に対応するノードは，4つのタイプ

テーブルタイプ

フロータイプ

リーフタイプ

因果タイプ

に分類される。したがって，ノードの各タイプに応じた調査を実行する機能が必要である。

2.3.3 付加機能

(1) フォールトツリーの表示

ビットマップディスプレイ上にフォールトツリーをグラフィカルに表示する機能である。実行済みの調査ノードと未実行のノードを区別して表示させる。

(2) 説明機能

どのような，経過を経て現在の調査ノードへたどり着いたかを示す機能である。

付録 2. 各ツールのマニュアルレベルでの特徴

（付録２）各ツールのマニュアルレベルでの特徴

ＡＲＴ（Ver2.0），ＫＣ（Ver2.0），ＫＥＥ（Ver2.1）の各シェルについて知識表現能力をマニュアルをとうして解析した結果を報告する。

ＡＲＴ（Ver2.0），ＫＣ（Ver2.0），ＫＥＥ（Ver2.1）の各シェルの知識表現手法はスキーマ（フレーム）を用いたデータ表現による階層関係と属性の継承と，それらを知識データとし，その上のルール記述や，オブジェクト記述が実行できるなどの類似した部分が多い。このタイプのツールはハイブリットタイプのツールとよばれ，異なった知識表現手法を複合して活用できる。

3.1　静的なデータの表現・・・データベースの階層と継承。

ＫＥＥ：　データの最小単位はユニットで与えられる。

各ユニット間の階層関係はsuperclass（親ユニット），subclass(部分ユニット)，members(インスタンス）により与えられる。slotはそのユニット固有のスロットを与えるOwnslot とメンバーに継承を与えるMemberslotに区別される。

Memberslotは，そのMembers ではOwnslot となり，親からの継承関係をファセットを用いることにより制御することができる。

ＫＥＥでの継承関係を与える”関係”は固定されており，ユーザ定義はできない。

ＡＲＴ：　データ最小単位はファクトであるが，スキーマ表現が許されており，データのコンパイル時にはすべてファクトにおとされる。スキーマ間の継承を与える関係”リレーション”及び，”スロット”もすべてスキーマで表現されており，従って，ＫＥＥにおけるファセットの概念はＡＲＴでは，そのスロットを持つスキーマとは独立に表現される。

システムが与えるリレーションは，継承を与えるもの，包含を与えるもの，インバースを与えるものの３種類である。インバースは逆リレーションの自動的な設定を行なう。包含を与えるリレーションは，それ自身の継承を引き起こすことはしないが各ＰＲＯＴＯＴＹＰＥに対して継承を引き起こす。ユーザは自分でリレーションを定義することができる。その継承の有無，新しく他の継承を引き起こす等の動作（new-relation，transitivity）を設定することができる。

システムが与えるリレーションはkernel-schema として定義されており参照，変更が可能である。

ＫＣ　：　この４つのシェルの中では最も豊富でデータ構造を与える。データの単位はスキーマである。スキーマ内部のスロットはＫＥＥとＡＲＴのどちらの立場（スキーマとして独立させる場合と，与えられたスキーマに依存する場合）にも与えることができる。

スキーマとしてスロットを独立させる場合は，そのスキーマをスロット表現

スキーマと呼び，ＡＲＴの場合と類似する。

スロットをそれを与えるスキーマに依存して考える場合はＫＥＥにおけるファセットの概念はメタスロットとして表現されている。

3.2 動的な表現

3.2.1 推論制御

ＡＲＴ： プロダクション，システムにおける前向きの推論と，それを補助する後ろ向き推論とが与えられている。前向き推論は通常どおりである。後ろ向き推論は，前向き推論の条件部がfactと照合されず，更に，同じリレーションをゴールパターンとして持つ後ろ向きルールが存在する時に実行される。後ろ向き推論が実行されるとその仮説がゴールとして登録され，それを検証するために，後ろ向き推論が開発される。この後ろ向き推論は，前向き推論の条件部における”デモン”的な役割を与える。即ち，もし必要とあれば後ろ向きルールを機動し，その結果，”ユーザに対して，質問する”などの処理が実行される。ＡＲＴの本領は，推論制御そのものよりも，次に述べる推論にともなって変化するデータ・ベースの構築にある（View Point）。ＡＲＴはオブジェクト指向風のメッセージの伝達，デモンの使用はできない。

ＫＣ ： ＫＣではスキーマ以外の推論制御構造はあまり重要視されていないようである。

ＫＣのデータ表現（スキーマ）上に既存の推論構造であるＯＰＳ－５，Prolog，およびメッセージのやりとりを行なうための特別の命令が用意されている。

ＫＥＥ： ＫＥＥの推論制御はオブジェクト指向型のメッセージとメソッドの起動による制御および，プロダクションシステムとしての前向き，後ろ向き推論が可能である。ただし，前向き，後ろ向きルールで記述されるデータ構造は，データ・ベースとのマッチングという意味では，Tell and Ask表現で与えられなければならない。

Tell and Ask表現はシステムにより与えられたフォーマット以外にユーザが定義可能ではあるがＬＩＳＰの世界で記述する必要があり面倒である。

推論制御における評価は，順序づけをするのが難しい。プロダクション・システムとして前後推論をうまく組み合わせているのはＡＲＴだが，メッセージ，メソッド，デモン等のオブジェクト指向タイプの制御は持たない。一応すべての機能をもつのはＫＥＥであるが，複雑なプログラムを書こうとすれば，Lisp記述が必要になる。

ＫＣにおけるＯＰＳ－５はもちろんだが，ＡＲＴのルールとfactのパターン・マッチングもReteアルゴリズムによる高速化が与えられている。

3.2.2　データ・ベースの分割およびブラックボード的な世界の記述能力。

ＫＣ，ＫＥＥ（Ver3.0）：　コンテキスト（ワールド）という概念がある。これはＡＲＴにおけるレベル－１のview-pointに対応する。ただし，ＡＲＴのように”世界”の推移を与えるよりも，旧データの保存を意味する場合が多い。親－子の間でのコンテキストのマージが可能である。この時，マージされた後のコンテキストが新しい親コンテキストとなる。これはＡＲＴにおけるview-point機能のbelieve である。

ＡＲＴ：　アートの特色はこのviewpoint 機能にある。

viewpoint 機能は次の視点から”世界”（factの集合）を構造化する。即ち，多重世界の表現，世界の変化の表現，および仮説世界の表現である。これらの世界の表現は（インプリメントとしては）データ・ベース上のfactにextentと呼ばれるインデックスを付加することによって実現される。多重世界では内側の世界が外側の世界を，世界の変化では，後で生まれた世界が初めの世界の事実を継承する。

これらの世界は推論部が実行され，しかるべきルールが起動されることにより動的に変化する。世界の変化はsprout，merge，poison によって，現在の視点（view point）をもとにコンテキストを生成，マージ，削除する。マージは通常システムが自動的に行なう。これは，探査空間の枝刈りを行なうことによりデータの発散と推論の緩慢さを除く。

さらに，仮説空間を併用することによって，推理的な整合性を与えることができる。例えば，仮説の一意性を与えるために，コンテキスト間の継承関係内

では同じ仮説はたてられない。また，同じ仮説をもつコンテキストは存在できないなどである。

この”世界の記述”ではＡＲＴの機能（view point）が際立っている。しかし，view point 機能をどのように用いるかは，まだ明確にされていないようである。ＡＲＴにおけるview point機能は黒板モデルの拡張と一般に言われているが，むしろＨＥＡＲＳＡＹ－ＩＩＩの（黒板モデルとしてではなく）コンテキスト機能の拡張と考えられる。

あ と が き

- The difficulty of evaluating expert system tools is a good sign.

by Clancy, W.J., and Rothenberg, J. "Evaluating Expert System Tools."
from Tutorial of the 6th AAAI (July, 1987).

第2部では，まず，1章で，エキスパートシステム開発ツールの評価項目，性能評価のために設定した小規模なテスト問題を提案し、その基本的な考え方について述べた。ついで，2章では，2つの標準問題について解説を行い，これを代表的な3つのツールでインプリメントした結果について報告した。

ただし，本報告では，時間的な制約から，評価用の枠組みを提案するとともに，ひととおりのインプリメンテーションにしたがった評価を示すにとどまっている。すなわち，各評価項目に十分な説明をつけること，この項目にしたがって既存のツールの評価を実施すること，また，性能評価用テスト問題の実現作業も行うことなどが課題として残っている。今後は，評価項目のうち，特に，基本的なタスクに関連する項目について開発方法論との関連を明確にするとともに，性能評価用テスト問題，標準問題に対する評価作業を継続していく予定である。その際の方針は次のようなものである。

・提案した評価項目にしたがって，各種のツールの比較を実施する。その際には，抽象的な性能評価用テスト問題のソフトウェア開発を通じて，実際にデータを収集するとともに，初心者によるツールの利用効果を測定評価する。

・標準問題をさらに分析して基本タスクとしての性格を明らかにする。そして，この面で各ツールの機能をどう使ったか，なぜ使わなかったかを調査するとともに，標準問題の実現に際し各種機能がどれだけ有効であったかを評価する。

なお，本報告で述べた実際に則した標準問題の設定にあたっては，レンズの骨組み設計問題についてはキャノン株式会社殿の，電気設備のトラブルシューティング問題については新日本製鉄株式会社殿の全面的な協力が不可欠であった。両社の関係者の方々に心よりお礼申上げる次第である。

あ と が き

- The difficulty of evaluating expert system tools is a good sign.
by Clancy, W.J., and Rothenberg, J. "Evaluating Expert System Tools."
from Tutorial of the 6th AAAI (July,1987).

第２部では、まず、１章で、エキスパートシステム開発ツールの評価項目、性能評価のために設定した小規模なテスト問題を提案し、その基本的な考え方について述べた。ついで、２章では、２つの標準問題について解説を行い、これを代表的な３つのツールでインプリメントした結果について報告した。

ただし、本報告では、時間的な制約から、評価用の枠組みを提案するとともに、、ひととおりのインプリメンテーションにしたがった評価を示すにとどまっている。すなわち、各評価項目に十分な説明をつけること、この項目にしたがって既存のツールの評価を実施すること、また、性能評価用テスト問題の実現作業も行うことなどが課題として残っている。今後は、評価項目のうち、特に、基本的なタスクに関連する項目について開発方法論との関連を明確にするとともに、性能評価用テスト問題、標準問題に対する評価作業を継続していく予定である。その際の方針は次のようなものである。

・提案した評価項目項目にしたがって、各種のツールの比較を実施する。その際には、抽象的な性能評価用テスト問題のソフトウェア開発を通じて、実際にデータを収集するとともに、初心者によるツールの利用効果を測定評価する。

・標準問題をさらに分析して基本タスクとしての性格を明らかにする。そして、この面で各ツールの機能をどう使ったか、なぜ使わなかったかを調査するとともに、標準問題のの実現に際し各種機能がどれだけ有効であったかを評価する。

なお、本報告で述べた実際に即した標準問題の設定にあたっては、レンズの骨組み設計問題についてはキャノン株式会社殿の、電気設備のトラブルシューティング問題については新日本製鉄株式会社殿の全面的な協力が不可欠であった。両社の関係者の方々に心よりお礼申上げる次第である。

昭和６２年９月発行

発行所　財団法人　日本情報処理開発協会
東京都港区芝公園３丁目５番８号
機械振興会館内
電話（０３）４３２－９３９０

６２－Ａ００２