

# ルーチン設計問題向け知識システムへの 制約指向技術の適用についての検討

永井保夫\*

本論文では、設計問題向け知識システムにおける制約指向技術について、ルーチン設計問題を対象にして検討する。まず、設計システムについて要求される一般的な機能について考察する。そして、設計過程モデルを明確にするために、設計対象についての分類を行い、基本となる設計タスクについて述べる。次に、上記のなかで特に、様式化の図られている設計過程により問題自身をwell-definedかつwell-structuredな問題として取り扱うことが可能である設計をルーチン設計問題と定義する。そして、より詳細なルーチン設計問題の分類を行い、実際の設計対象領域と対応付ける。最後に、ルーチン設計問題を対象とした知識システムに要求される機能と制約指向技術の適用について検討する。

**キーワード：**制約，知識ベースシステム，ルーチン設計，制約指向技術，合成型アプリケーション

## Towards Application of Constraint-Based Processing Technologies to Knowledge-Based Systems for Routine Design Problems

Yasuo NAGAI

This paper reports on the framework of the knowledge-based system architecture for the routine design problems and necessary functions, while defining the routine design problem and classifying types of these problems. Especially, it focuses on and specifies application of constraint technologies to problem solving, consisting of constraint representation and constraint processing.

**keyword :** constraint, knowledge-based system, routine design, constraint technologies, synthesis application

### 1. はじめに

現存する知識システムの適用対象は解析型問題と合成型問題の二つに大別される。合成型問題は解析型問題とは逆にある仕様を満足する対象を生成していく時に、既知の構成要素を組み合わせることにより未知の対象を生成するものとみなされ、問題解決時にシミュレーションなどの解析を実行するタスクが内在する。これは、設計自身の有する試行錯誤性により設計対象

が動的に変化していく、つまり、動的に対象表現モデルを取捨選択し、評価していくためである。そのため、設計プランを考慮した知識システムのアーキテクチャの検討が必要となる。従って、設計過程における各々の対象の概念関係は通常、階層的になるのでそのための表現方法が必要である。合成型問題を対象とした知識システムのアーキテクチャは解析型のそれほどは明確化がなされていない。しかしながら、最近、合成型問題の代表例である設計問題の中で特別なケースを対

\*東京情報大学総合情報学部情報システム学科

Tokyo University of Information Sciences, Faculty of Informatics, Department of Information Systems

象とした知識システムの研究が事例を主体として盛んになりつつある。本研究では、このような設計問題を対象にした知識システムのアーキテクチャの明確化ならびにそのために必要となる要素技術（特に、制約指向技術）を明らかにすることを目標とする。まず、はじめに知識システムを包含した設計システムに要求される一般的な機能について考察する。設計過程モデルを明確にするために、設計対象についての分類を行い、基本となる設計タスクについて述べる。そのためには、設計対象自身が設計システムとして定式化され取り扱いが可能であることが必要となる。次に、上記のなかで特に、様式化の図られている設計過程により問題自身をwell-definedかつwell-structuredな問題として取り扱うことが可能である設計問題をルーチン設計問題と定義する。さらに、より詳細なルーチン設計問題の分類を行い、実際の設計対象領域と対応付ける。最後に、ルーチン設計問題を対象とする知識システムのアーキテクチャ実現に要求される機能と制約指向技術について検討する。そのために、特にルーチン設計問題における設計過程のモデル化を考慮した場合の制約表現に基づいた問題解決機構に焦点をあてて説明する。

## 2. 設計システム

ここでは設計システムとは、CAD (Computed Aided Design)・CAM (Computer Aided Manufacture) システム及びDA (Design Automation) システムのことを総称して示している。具体的な設計の対象は非常に多岐にわたり、さらに各対象によりシステムの適用形態もかなり異なる。設計を対象とした知識システムの考察をするためには設計システム全体の概観の整理が必要であり、これにより設計システムにおける知識システムの占める位置付けを明確化する。

### 2.1 必要とされる機能

設計とは、定義可能な言語、記号及び図形によって表出可能な概念を要求として与え実際に存在する人工物に変換するような創造的な知的行為を実現するものである。もう少し情報处理的立場よりみると、要求を与えて、その要求を満足するモデルの構造・形状の創造を行う行為である。本節では、特にこのような設計行為を合成タスクと解析タスクという2つの行為を明確化する方向から眺める。さらに、設計を概念設計、

基本設計、詳細設計という3つのフェーズから構成される設計過程を基本にして各フェーズで必要とされる機能について示し、解析タスクとモデルについての対応についても考察する。なお、各フェーズでの設計はモデルを生成し、解析・評価、修正という試行錯誤的なタスクを実行する。

概念設計は、これから設計しようとしている対象（人工物）についての要求の列挙（記述）、仕様定義などを包含した解決されるべき問題の絞り込み及び定式化などから構成される。このフェーズは設計すべき対象に対する概念や適用するアイデアを求めるという創造的な行為から始まる。一部の様式化された設計を除くと大部分が合理的に定式化がなされてはならず、どちらかという設計者の有している（頭の中に存在している）スキル（思考や経験）により行われている。つまり、モデルは設計者が頭の中に有しており、そのモデルの取捨選択・評価がなされる。

基本設計では、概念設計での結果をうけて設計対象を詳細化するための様々な視点によるモデルの生成、解析・評価、修正がなされ、いくつかのモデルの選択と絞り込みが行われる。この場合の解析・評価手法は各モデルにより決定されるが、解析手法の確立されていない場合には実験やシミュレーションにより決定することも多い。

詳細設計は、基本設計の結果に基づき、絞り込まれたモデルにより設計対象の形状や構造、要素間の関係などを詳細化して決定するものである。その際のモデルに基づいた最適化や評価も合わせて行われる。

## 2.2 設計過程モデル

### 2.2.1 一般設計論における設計過程モデル

ここでは、設計過程モデルを論ずるために一般設計論について紹介する。設計過程モデルを考えた場合には、設計仕様・要求仕様、機能、（構成）要素、構造などの概念とそれらの概念間の関係の定義が不明確なので、一般設計論で言われている定義について述べる。

まず、要素と構造の関係について説明する。要素は、様々な性質を持っており（つまり、属性によって表現され）、ある視点により眺めたときに、固有に表われる属性によりふるまい（機能）が観察されるものである。要素は様々な視点（例えば、機械的、電氣的、流体的など）により有する機能は異なる。例えば、機械

的要素は得られる機能は自由度の大きい形状によって生じるために、電氣的要素（たとえば、電気素子）ほど抽象表現が容易には行えない。つまり、電氣的要素であれば、電氣的な視点だけに限定した電気回路法則に従った形で機能を表示することは比較的可能であり、要素と構造の分離も容易である。システムは、複数の要素から構成され、かつ要素間に接続関係が定義されることにより機能を生ずる。この場合の構造とは要素の集合と要素間の接続関係により定義される。つまり、要素と構造という概念は機能を有したシステムを実現するために深い係わりを持っていることになる。機械システムの場合には要求される機能を得るために要素と構造とを表現することが必要であり、具体的には要素を部品、接続関係を部品間の接触関係とそれぞれみなすことでシステムの実現を図っている[53]。

設計過程は、設計仕様・要求仕様を入力として、設計解を出力するプロセスとみなしモデル化される。これは設計仕様・要求仕様（設計対象に必要となる機能）をより詳細化した機能の組み合わせとして表現し、それぞれに対して要求される機能を実現する方法（方式）を明確化して選択することであり、設計仕様を詳細化していくことに相当する。機械設計では設計仕様、つまり要求機能を満足するために設計対象が複数の機械要素から構成される形で機能分割をおこない（設計の内包化に相当）、分割された機能を実現するような機械を実現する方法を明確化し（設計の外延化に相当）、さらに選択・評価を繰り返すことで機械要素を詳細化していき、最終的には部品設計に到達し全体の設計が完了する。設計仕様とは設計対象が持つべき性質を抽象的概念により表現したものである。なお、上記の設計過程モデルでは設計仕様の機能分割方法が問題となる。これは設計仕様、すなわち要求機能が既知の物理現象の組み合わせにより表現できれば、それらの物理現象を実現する構成要素（機能ブロック）の組み合わせで実現可能である。しかしながら、機能を実現するのに、物理現象が明確化され、その物理現象に基づいた形で機能を生じる要素が機能要素として与えられればよいが、その機能が物理現象に基づいて実現できない場合も存在する。その場合は、機能を実現する要素を人手によって実体化し、解析作業をへて実験式などを求めることで構成要素の諸機能を定義すること

が要求される[54][55][49]。

次に、設計過程モデルは、以下のように分類される[54]。

1. 仕様の機能表現から設計解への対応関係が与えられている
2. 設計解に近づくような実体概念を次々と提案する方法が与えられている

1を対応型、2を収束型と呼ぶ。さらに、詳細なモデルの例について示す。

#### 1. 対応型

##### (a) 全数対応モデル

設計仕様から設計解への対応関係が与えられ、与えられた仕様に対しては必ず解が求まる。つまり、設計過程は検索問題とみなされ、設計解は予め設計された結果を検索することに相当する。

##### (b) 計算モデル

設計仕様と設計解の間に写象関係が与えられた時に、与えられた写象関係を関数とみなし設計仕様を入力として適用することで設計解を出力することに相当する。

##### (c) 生成モデル

何らかの生成規則があつて、設計仕様にその規則を適用することで設計解がアルゴリズムに（記号処理によって）得られることに相当する。

#### 2. 収束型

##### (a) 範例モデル

このモデルでは、機能的に類似している実体（人工物）は類似した属性を有し、また、その逆も成立し、かつ設計者の実体概念は階層的であるということが仮定されている。設計仕様から設計解を求める場合に、すでに与えられた実体から仕様を満足するような実体を検索・選択する。しかしながら、それでは仕様が完全に満足されていない場合は、その実体の階層をさらに構成要素に分割し、しかもその幾つかの構成要素を既知の構成要素と置き換えることで最終的な設計を完了する。

### 2.2.2 設計主体と設計過程

設計システムを考える前に、まず、設計の主体が計

算機に比重のあるものなのか、それとも人間に比重のあるものなのかを考察する必要がある。

以下では、計算機に比重がある設計を人手設計支援用計算機設計、人間に比重のある設計を計算機支援による人手設計と分類して説明する [19] [27]。

## 1. 人手設計支援用計算機設計 (human-aided computer design)

以下の項目を有するケースが該当すると考える。

- ・ 概念設計における、要求定義や仕様記述が定式化されている
- ・ 各フェーズのモデルが明確化されている
- ・ 設計過程自身の様式化がなされている
- ・ 各フェーズのモデルが明確化され、その解析・評価手法の明確化されている

ここでは、well-definedかつwell-structured design problemとして定式化がなされている。つまり、設計過程自身の様式化が行われている設計を対象とする。その場合、概念設計における要求定義や仕様記述が記述言語という形式で定式化され、各設計フェーズにおいて使用されるモデルとそのモデルに対する解析・評価手法が決定されている場合には、自動設計が行われ、DAシステムという形態で利用される。また、DAシステムと同様に概念設計での仕様記述が定式化され、かつ各設計フェーズでのモデルは明確化されているが、そのモデルの解析又は評価手法が定式化されずに、一部を設計者が代用して実行する（主に、解析手法が与えられて、その解析情報に基づいて人間が評価を行う）ようなインタラクティブ設計もこの範疇に属する。

## 2. 計算機支援による人手設計 (computer-aided human design)

以下の項目を有するケースが該当すると考える。

- ・ 概念設計における要求定義や仕様記述が不完全である（定式化して与えられない）
- ・ 各フェーズでのモデルが明確化されていない
- ・ 設計過程自身の様式化がはかられていない
- ・ 各フェーズでのモデルの明確化はなされてはいるが、その解析・評価手法の明確化が行われていない

つまり、この場合の設計は、ill-defined and/or ill-structuredな問題で、設計過程自身の様式化（定式化）が図られていない。計算機は、設計の流れの情報管理や設計者が意思決定（decision making）をするのに必要な情報の提供を行う。この場合、設計という知的行為の主体は人間であり、計算機はあくまでもそのためのサポートとして利用する。CADシステムの大部分がこの範疇に属する。

ここでは、特に、1の人手設計支援用計算機設計のようなwell-definedかつwell-structuredな（様式化された）問題を対象とする設計をルーチン設計と定義して検討を行っていく事とする。

## 3. ルーチン設計問題

### 3.1 ルーチン設計問題の定義

ルーチン設計問題は、以下の項目が満足される設計問題である。

- ・ 機能表現又は動作記述が要求定義や仕様記述という定式化可能な入力形式で明確化された時、その入力を構成要素とその接続関係という出力形式へ分割または写象する方法が決定されている。つまり、仕様表現が入力として与えられた時、その入力に対してインデキシングされたモデルに基づいて詳細化を行うことで、構造表現や形状表現という出力形式へ写象、または変換するための方法が既知である。
- ・ 各設計段階において、設計プランが明確化され定式化可能である。つまり、入力である仕様表現を機能分割問題や構造分割問題とみなした場合の問題分割法、詳細化法、解析法及び評価法などが明確化されて設計プランを構成する。

さらに、過去の設計例（以前に解かれた設計問題）と同様な経験による知識を利用した設計も同様にルーチン設計問題と定義することにする。入力として与えられる設計仕様は十分定式化の可能な（well-understood）ものであり、標準的な問題解決手法が存在している設計である。大部分のDAシステムはこのような意味ではルーチン設計問題とみなす事が可能である [41] [25] [5] [26] [45]。

それ以外に代表的なルーチン設計問題としては、修正設計や編集設計などが挙げられる。これは過去の設

計事例を改善していく方向（設計対象の設計パラメータの明確化とその修正方向（次元）の指定）で設計を行っていくものでパラメトリック設計の一種である。特に、機械設計においては顕著な例となっている [8] [28] [30] [38]。

機械設計を対象にする場合には、幾何学表現、特に、3次元情報や製造及び組み立て情報が非常に強く結びついているために、設計対象自身をモジュラー化する事が難しい。つまり、幾何学的特徴（形状記述）が機能記述や製造情報に非常に依存しているために、幾何学的特徴が変化すると対象の動作もそれに伴い変化するからである。これは動的に設計対象の構成要素、例えば、素子や部品のモデルを生成することに相当する。そのため、回路（LSI）設計とは大きく異なり、対象の振る舞いや機能に対して、構成要素の抽象化を図る事が困難である。従って、要求仕様を与えても、その通り動作するかを決定する事が難しく、解析タスクが必ず必要となる。このように、形状表現を入力仕様として与えることが設計において重要となる場合には、ルーチン（自動化または半自動化）化する事は難しく、あくまでcomputer-aided human designという方向でCADツールを用いて設計を進めていく必要がある。

回路設計 [4] [7] [25] [45] は、設計自体の階層化がなされ、特に各設計レベルの抽象化が行われていることが多いので、独立してトップダウンな方法で設計がなされている。機械設計と比較するとかなり異なり、幾何学的特徴、たとえばレイアウトレベルは機能表現に対しては余り影響を及ぼさない。設計対象を仕様表現として定式化して入力する方法は、専用のハードウェア記述言語を用いて、設計者が十分定式化可能な形式で行う。例えば、シリコンコンパイラ [24] [42] のようなDAシステムでは、設計対象の動作記述、アルゴリズム記述、または、機能記述を入力して、幾何学的表現を出力する。その場合、各設計レベルにおいて、設計の性能評価用に、シミュレーション及び検証用関数が与えられた場合の組み合わせ最適問題とみることができる。

回路設計は、入力仕様が示しているふるまいを満足するように構成要素の組み合わせを求める問題とみなせる。構成要素はその機能が抽象表現されたもので、I/O関係（入力・出力）が厳密に定義されており、その組み合わせと構成要素とそのI/Oの接続関係を決定

することに相当する。接続するということは、I/Oでの信号が同じ状態（値）になるという強い制約を意味する。但し、この場合は設計要求及び制約や評価関数におけるトレードオフの取扱いが非常シビアな問題となる。従来の手続き型アルゴリズムにおいてヒューリスティックスを用いたほうが有効な場合のみ、知識システムとして置き換えることが可能である。DAシステムにおいては、ある設計をするのに用いられる構成要素や分割されるモジュールなどを決定するアーキテクチャに関する知識、構成要素をインスタンスーションするモジュール選択用知識などをルール表現に置き換えているのが通常である [52]。

### 3.2 ルーチン設計問題の分類

新たに、設計に対して新規設計、組み合わせ設計、パラメトリック設計という3種類のレベル [48] を設定する。

新規設計では概念設計、基本設計、詳細設計などの各設計段階において、入力仕様が与えられた場合に、過去に行われた設計結果を利用せずに最初から新たに設計を行う。これは設計者の創造的行為により設計がなされたり、設計過程自身の様式化がほとんど行われていないことに相当する。

組み合わせ設計では、基本設計や詳細設計などの設計段階において、入力仕様を満足するように、過去に行われた設計結果（これを基本要素とみなす）を組み合わせで設計を行う。

パラメトリック設計では、詳細設計段階において以前になされた設計によって設計対象の構造や構成要素が詳細化され属性によって表現されている場合に、属性のパラメータだけを指定・修正することで設計を行う。

ここでは設計システムとして定式化が可能な設計（組み合わせ設計とパラメトリック設計）を対象として、以下のようなルーチン設計問題の分類を行う。

#### 1. 組み合わせ設計

##### (a) トップダウン的アプローチ（ケース1）

- ・設計仕様: ハードウェア記述言語たとえば、アルゴリズム記述、または動作記述や機能の仕様記述など
- ・設計解: テクノロジーに依存しないハードウェア

構成要素（レジスタ、演算器。データバス、制御信号など）のリスト。つまり、データバス回路やさらにその下位レベルの論理回路。

ケース1は回路合成問題に対応する。設計仕様に対するテクノロジーに依存しないハードウェア構成要素の割り付け問題とみなすことが可能である。これは言語変換問題と考えられるのでコンパイラで用いられている手法〔1〕を適用することである程度対処可能である。つまり、設計仕様に示されているデータフロー情報や制御フロー情報などが論理的に保持される形で、段階的にハードウェア構成要素を割り付けていくことにより設計解への変換が行われる。しかしながら、実際には複数の設計形式やコスト、パワー、スピードなどの制約を満足させる複雑な問題となっている。従って、コンパイラ手法以外にも制約を満足させるようなトレードオフ問題を考慮する必要があるが、基本的には段階的詳細化問題、つまりトップダウン的な戦略による組み合わせ問題とみなされる。なお、ハードウェア構成要素は抽象的又は機能的構成要素に相当する。このケースは自動化システム形式ではシリコン・コンパイラとして実現が図られている。

但し、デジタルシステムでは、表現（記述）レベルが以下のように明確な階層化されている。

- ・システム動作記述レベル
- ・機能記述レベル
- ・論理記述レベル

なお、そのために問題となるのは（ある上位レベルの動作、または機能）記述を実現するための（下位レベルの）構成要素とその接続関係の組み合わせはいくつも存在するという事である。さらに、同記述レベルにおいても構成要素とその接続関係の組み合わせもいくつも考えられる（最適化が行われるため）。

#### (b) ボトムアップ的アプローチ（ケース2）

- ・設計仕様: 構成要素と構成要素間の部分的接続関係や性能などの制約
- ・設計解: 設計仕様及び制約を満足する構成要素の配置決定

ケース2は、構成要素・部品の配置問題やスケジューリング問題、たとえば、計算機構成機器の配置〔26〕〔5〕、ジョブ・ショップスケジューリング問題〔21〕などが対応する。ここでは特に、構成要素・部品の配置問題について述べる。構成要素の集合を設計仕様と

して与えると、その中の制約を満足させながら構成要素間の相互関係を定めつつ設計対象の構築を行っていくものである。その設計対象の構成は全体ではなく、部分的に構成要素ごとに構成可能になっている。すなわち、構成済みの構成要素を組み上げていくことにより全体を構成する。構成要素・部品の配置問題でのタスクは構成要素選択サブタスク、および構成要素組み合わせサブタスクからなる。構成要素選択サブタスクは、すでに得られている構成要素の性能特性や入手容易性といったテクノロジーに依存したハードウェア構成要素を考慮して行われる。構成要素組み合わせタスクは各構成要素が他の構成要素と関連してどのように配置されるかを決定する。そのとき決定される構成要素とそれらの接続関係、つまり構造表現はトポロジカルな関係、幾何学的関係、空間的關係をそれぞれ考慮したものがあり、空間的關係を考慮した配置が最も困難な問題である。ケース2は、ケース1と比較すると生産・製造（実装）段階に深く関連しており、構成要素間の接続関係が強い制約として働いており構成要素選択がなされるたびに制約が動的に変化するためボトムアップ的アプローチであるといえることができる。構成要素でケース1とケース2を比較してみると、前者は構成要素が抽象的、又は機能的であるのに対して、後者は構成要素が物理的に存在するものである。

#### 2. パラメトリック設計（ケース3）

- ・設計仕様: 要求機能仕様、及び要求性能仕様
- ・設計解: 構成要素の詳細化。つまり、ユーザの要求や制約を満足するような構成要素の属性値の決定

ケース3は設計対象の構造表現（構成要素集合、およびそれらの接続関係）が複数の設計プラン（案）として与えられ、さらに構成要素についても属性表現として明確化されていることが仮定されている。さらに設計案に基づいた構造表現を解析・評価する方法がわかっている設計である。従って、構成要素属性のパラメータ決定問題とみなして定式化することが可能である。この場合、構成要素の標準部品による実現と非標準部品による実現の可能性が考えられるため、その属性に対してとれるパラメータ値には次の2種類のタイプが考えられる。

- ・離散量（標準部品）
- ・連続量（非標準部品）

構成要素は部品の属性決定を行う関係表現、設計式などを含んでおり、構成要素間の接続関係は全体一部分関係を含む。なお、DSPLでのAIR-CYLシステム[8]やPRIDEシステム[28]が対象としている設計はこの分類に属する。

### 3.3 ルーチン設計問題向け知識システムのための制約指向技術

ルーチン設計問題向けの知識システムを開発していくためには設計対象に依存した知識が必要とされる。従って、設計問題固有の知識表現と、その知識を用いた効率の良い問題解決の方法が設計システム研究におけるキーポイントとなっている。制約という概念を導入することにより、設計知識を宣言的な形式で容易に表現でき、解空間を規定することで余分な探索を減らし問題解決能力を向上させることが期待される。本節では、このようなルーチン設計問題向けの知識システムのための知識表現ならびに問題解決において適用すべき制約指向技術について説明する。

#### 3.3.1 制約表現

##### 1. 制約表現

ここでは、制約とは適用対象の構成要素間での成立する関係(条件)、その性質間で成立する関係(条件)、及び満足されるべき法則(規則)などを表現したもの(その表現形式としては等式や不等式など)であると定義する。例えば、対象モデル表現より得られる構造情報は陽に表現された制約である。さらに、具体的には、回路解析におけるキルヒホッフの法則やオームの法則、ジョブショップスケジューリングにおけるリソースの数量、コスト、オペレーションの優先順位、期日など、そして線画理解における物理的に可能なエッジの接続関係などがあげられる。しかしながら、従来、与えられた制約が必ずしも有効に利用されているとはいえない。そこで、このような制約を有効に利用することにより、解空間における探索に制限を与える事が可能となり、余分な探索を行わないで効率を上げることが期待できる。従来の知識システムには制約という概念を明示的に表現可能な環境を提供しているものは余り多くない。システムの構築者自身がツールの開発言語を駆使して適用対象に依存した制約表現の適用メカニズムの実現を図っているのが現状である。ところ

で、設計問題を考えてみると制約を満足させる問題に帰着可能な場合が多い。たとえば、生成・検査方式では検査部分で制約をチェックすることに相当する。又、段階的詳細化方式では仕様を抽象レベルからより具体レベルへ詳細化していくもので、各レベルごとに適用される制約も動的に変化していく。一般に、与えられた仕様を詳細化していく場合に各部分問題への分割化を行なうのが普通であり、さらに分割された部分間には制約同士のインタラクションが存在することが多い。

このような設計やスケジューリングなどの合成型問題を対象にした場合の制約の分類化とその有効な適用による問題解決に必要な機能について述べる。

##### 2. 制約の分類

制約をルーチン設計問題には適用されるが一般的な場合と、ルーチン設計問題に依存した場合の2つに分けて考察する。

###### (a) ルーチン設計問題に適用される一般的な制約

ルーチン設計問題に適用される一般的な制約について以下の項目によって分類した。

###### i. 生成方式による分類

生成方式による制約には、静的な制約と動的な制約が考えられる。静的な制約とは、予め静的に与えられた制約であり一定で変化しないという不変性が保持されている。動的な制約とは、ユーザによるインタラクションやシステムより制約が与えられることで、静的な制約とは逆に状況に応じて適用範囲や制約自身が動的に変化(追加や削除)するものである[40]。これは不完全な知識と解釈可能であり、制約の変化による知識ベース中の信念の変化の管理を行うため、TMS (Truth Maintenance System) [18] 的な機能が必要となる。

###### ii. 重要性による分類

重要性による制約には、義務的な又は必須な制約と示唆的制約が考えられる。その場合、すべての制約が対等なものとして選択・実行されるわけではない。つまり、その重要性はコンテキストや時間などの概念に依存する。義務的な制約はすべてが満足されねばならぬもので、一般的には陽に与えられるものである。示唆的な制約は弱い制約

とも言われ、選択枝より最良の枝を選択するためのガイドとして使用される。これはルール形式で表現可能なものもあり、プライオリティなどの重み付けが行われることが多い。

### iii. 適用範囲の限定による分類

適用範囲の限定による制約には、ローカルな制約とグローバルな制約が考えられ、探索空間における状態の評価に使用される。ローカルな制約はあるモデル、オブジェクト、プロセスなどの中で状態が変化するものに対して探索を行うために使用し、有効範囲もそのなかで閉じているものである。グローバルな制約は、ある状態の評価をするために、ローカルな制約だけを適用するのではなく、適用範囲を限定しないで関連するすべての制約を適用して評価をするものである。例えば、これは解空間を分割化して探索していく場合には現在の状態に至るまでに関連した状態に適用されたすべての制約を考慮したり、分割された解空間同志の評価を行なうことに相当する。

### iv. 伝播される変数情報による分類

伝播される変数情報による制約には変数に対して、値を伝播する制約と値の取り得る領域を区間として伝播する制約が考えられる。現在、制約論理プログラム [15] [23] やCONSTRAINT [46] に代表される制約表現システムでは、値を伝播する制約だけを取り扱っており、value inference又は、expression inferenceと呼ばれている [13]。値の取り得る領域を区間とし伝播する制約は、不等式として表現された制約が変数をコンスタントな値ではなく区間集合とみなして伝播するものである。設計問題を考えた場合、従来のオペレーションズリサーチを利用して解を求めるような問題を含んでいる場合が少なくないので、区間集合による制約伝播という統一的な枠組みで取り扱う機能をアーキテクチャに対して考慮する事は非常に重要であると考ええる。

制約の分類を行ったが、実際にはひとつの制約が上記の機能項目を複数有しているものと考えられる。但し、現段階では以上の項目によりすべての制約表現を分類するには不十分であり、更なる検討が必要である。

次に、制約問題について従来より行われている研究に基づいた分類を行う。制約問題とは、与えられた制

約を満足するように変数集合に対して値を割り付けるような問題である。制約問題については、主に、E Lシステム [44] 及び制約言語研究から端を発したものと線画多面体解釈用のエッジラベリング [12] 及びフィルタリングによるラベリングアルゴリズム [51] 研究から端を発したものがある。前者は制約を代数式とみなしその代数式の変数に対する入力が値として与えられた場合、代数演算による関数として解釈がなされて出力変数に対する値が決定されるような制約問題を対象にしており、制約プログラミング言語として研究がなされてきた。後者はパズル、グラフ同形判定、グラフラベリング、巡回セールスマン問題などを対象にした整合ラベリング問題 [39] または制約充足問題 [14] [20] という形で研究がなされている。制約充足問題 (CSP) は変数の有限集合が与えられ、かつその各変数に対する値が有限ドメイン集合として与えられた場合に、変数間に成り立つ関係、つまり制約を満足するように各変数に対して値を割り付けると定義される。制約問題の研究といっても両者の例をとってみただけでも非常に異なったアプローチをとっており、当然、制約表現や解釈とその適用メカニズムも異なる。我々はルーチン設計問題においても上記の制約問題研究との関連をできるだけ考慮する方向で検討を行っている。

### (b) ルーチン設計問題でドメインに依存した制約

設計、特にルーチン設計問題でドメインに依存した制約について説明する。設計における制約は様々考えられる。そこで概念設計、基本設計、詳細設計からなる単純化された設計過程と関連づけて考察する。概念設計では、要求定義・仕様記述から得られる制約、特に性能やコストなどが制約として考えられる。基本設計では、制約は機能記述を実世界へ写象するために必要となる物理法則や幾何法則を決定する環境が設定されることで得られる。つまり、性能を解析・評価するモデルが取捨選択が与えられた場合に、制約はそのモデルより導出される。詳細設計では、絞り込まれ定式化のなされたモデルに従って詳細化していくのであるが、その設計対象の形状や構造、構成要素の決定や要素間の関係などを決定するのに関連する知識も制約とみなすことができる。さらに、設計対象により適用するモデルも当然異なるのでそれに対する制約も異なることになる。たとえば、機械設計では構造力学や熱力

学、材料力学などに基づくモデル [30]、回路設計では物理法則（電気回路／電子回路法則）に基づくモデルなどが考えられ、それぞれより導出される制約も多様である。

特に、ルーチン設計問題では構造に関する制約に対して考慮する必要がある。構造に関する制約は設計形式や階層設計での抽象レベルでの仕様及び要求などにより反映されるものであり、構造決定、分割化、より下位（低）レベルでの設計形式の決定などを行う。この制約は設計が階層設計である場合、つまり低（具体）レベルの設計が行われる場合には制約自体の伝播が行われることに注意が必要である。設計形式に関する制約は設計対象の構造の決定やより低（具体）レベルへの問題分割方針の決定などを行う。このような構造や問題の分割によって、制約自体の分割（分配）化がなされる。論理回路合成では、様々な設計レベルにおいて選択された抽象デバイス（モジュール）を物理デバイスに置き換える、つまり実装方式についてテクノロジーに依存した制約が存在する。この制約はデバイスの実装テクノロジーに関するものが考えられる。

### 3.3.2 ルーチン設計問題向け知識システムで取り扱われる設計問題

設計問題向け知識システムでは、設計問題として設計対象の構造（機構）の探索問題、構造諸元（連続値）の最適化問題、構造諸元（離散値）の探索問題、設計対象の構造の変換・最適化問題という4つの問題ならびにそれらを組み合わせた問題が取り扱われていると考えられる [32]。設計対象の構造（機構）の探索は設計案によって設計対象の構成方式が与えられているか、もしくは与えられた構成要素を組み合わせで行われる。設計対象の構造（構成要素とその属性値）が決定された後の詳細化処理は、具体的に実現されるべき構成要素やそれらの数量などの実現条件を考慮した構造諸元（連続値）の最適化問題や構造諸元（離散値）の探索問題とみなされる。

たとえば、回路設計では予め与えられた構成要素を組み合わせることで設計対象の構造を決定してから、構成要素とその属性値の詳細化を行い、トレード・オフを解消するために構造変換などの最適化を図る。機械設計において設計対象の構造（機構）があらかじめ与えられている場合には、これらの検索によって仮定

された構造に基づいて、構造諸元（離散値）を探索したり、構造諸元（連続値）を最適化する問題として取り扱う場合が多い。特に、パラメトリック設計はこのような設計対象の構造（機構）が決定されている場合に、構成要素の属性値を決定する（最適化する）設計である。

### 3.3.3 ルーチン設計問題向け知識システムの知識表現

設計知識は、設計対象そのものに関する知識と、設計対象を解析・詳細化し設計解を求めるための知識に大別される。前者の設計対象に関する知識は設計対象の構造、形状、属性などを問題解決機構やユーザが処理可能な形式で表現したもので、従来から対象モデルと呼ばれており、設計対象の構造の可変／不変性や設計パラメータの離散／連続性をなどを考慮する必要がある。後者の対象モデルの解析方法や設計手順に関する知識をここでは設計手法知識と呼ぶことにする。すなわち、設計手法知識は対象モデルの利用法や設計パラメータの値の求め方、設計手順、解候補が複数あった場合の探索手順、設計要求を満たさない時の解の補正操作、および解の評価に関する知識などからなり、設計案の操作を実行するために用いられる。これらの知識には、関数や不等式で表されたもの、カタログや図表の検索に関するものなどがあり、さまざまな種類や表現形式を持つ。さらに、設計公式やカタログなどの汎用的な知識と、ある特定目的に対してのみ有効なヒューリスティクスが混在している [33]。設計問題向け知識システムの知識表現を考えるにあたっては、これらの役割や性質に応じた整理が必要である。特に、対象モデルと設計手法の知識の明確な分離が重要である。また、設計とはこの設計対象の表現モデルの取捨選択、修正、詳細化といった繰返し処理であるとみなされる。

知識表現に対する制約表現導入のメリットは、宣言的知識の取り扱いにより制約知識をさまざまな方法で表現かつ利用できることで、表現能力を向上させることができる。さらに、知識のインクリメンタルな追加や修正による知識の維持が容易になり、知識ベースをよりコンパクト化できるので、ユーザの知識の記述の負担を軽減できる。さらに、制約指向プログラミング言語（制約論理プログラミング言語 [2]）により、OR

で利用される数式モデルならびに示唆的制約や義務的知識を形式的に取り扱うことができる。

### 3.3.4 ルーチン設計問題向けの設計タスクと制約問題解決機構

#### 1. 設計タスク

ここでは、基本となる設計対象のモデル、すなわち構成要素とそれらの接続関係からなる構造表現、その解析手法ならびに評価手法が決定されているルーチン設計問題を対象として説明をおこなう。ルーチン設計問題は過去の設計例、すなわち以前に解かれた設計問題や設計時に得られた経験による知識を利用した設計であり、入力として与えられる設計仕様が良定義可能("well-defined")で、標準的な問題解決手法が存在する設計である。図1に示すようなルーチン設計問題での設計過程は、問題を部分問題へ分割して、設計が進む方向でモデル化され、各レベルにおける設計タスクがトップダウンに実行される。これらの設計タスクは、設計案の提案、設計案の解析・検証、ならびに設計案の評価・修正からなる設計作業により表現される [10] [33] [36]。

設計案の提示では、与えられた設計仕様からそれを実現可能な設計案が提案される。その実現手段としては、問題分割-解合成による方法、事例検索による方法、および制約充足による方法がある [10]。問題分割-解合成による方法では、設計仕様の部分集合をさらに小さい設計問題に分割される。事例検索による方法では、事例として過去の設計例を参照して設計解が求められ、その場合、標準化された設計結果を検索する方法とその部分的な修正を行う方法とに大別される。制約充足による方法では、設計過程を制約充足問題とみなして設計問題が解かれ、制約充足または最適化が行われる。設計案の解析・検証では、設計案を解析し、設計要求(制約条件)を満足しているかが検証される。ここでは、設計タスクに固有な解法が用いられ、シミュレーションにより対象とすべき特性が求められる。設計案の評価では、ある評価尺度に基づき設計案の評価をおこない、設計での失敗の原因を分析する。また、設計案の修正では、設計案の失敗に関する情報に基づき、設計要求を満足するように設計変更がおこなわれる。

以上のようなルーチン設計問題における設計過程

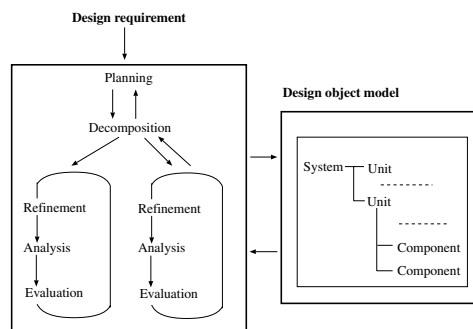


図1：ルーチン設計問題での設計過程モデル

は、設計仕様が与えられて設計解が生成されるまでの過程であり、設計空間での解の探索問題とみなせる。一般にこのような場合の探索空間は大規模であるために探索の効率化が問題となる。このような設計過程の制御を行うためには、設計タスクにおいて設計案の探索制御機能が必要である。

#### 2. 制約問題解決機構

ルーチン設計問題向け制約問題解決機構では、設計タスク(設計過程)に基づいた問題解決の実現が必要である。

設計タスクを実現する手法には、設計問題を問題空間探索とみなし探索をおこなう問題解決手法と解空間を探索せずに解を求めるアルゴリズムを用いた問題解決手法の2つ手法が考えられる [10] [36]。前者は主にAIにおける探索手法が用いられ、知識を用いて解空間を規定し、余分な探索を減らすものである。この手法は、生成・検査法、抽象化概念を取り入れた階層的生成・検査法 [50]、制約伝播や制約の能動的利用法である前方チェック法などを取り入れた生成・検査法 [50]、プロダクションシステムなどを用いて実現されている [36] [34]。後者は数値計算ならびに記号計算に関する手法やOR(最適化問題としての)手法が用いられ、たとえば、代数方程式の数値解法により設計パラメータを求めるものがひとつの例である。また、拘束条件解法理論もこの手法の一種である。

図2に示されるルーチン設計問題向け制約問題解決機構では、これらの手法において制約知識を用いることにより、設計問題の解または部分解のチェック、部分解の生成、解の修正、推論結果の伝播、推論結果の

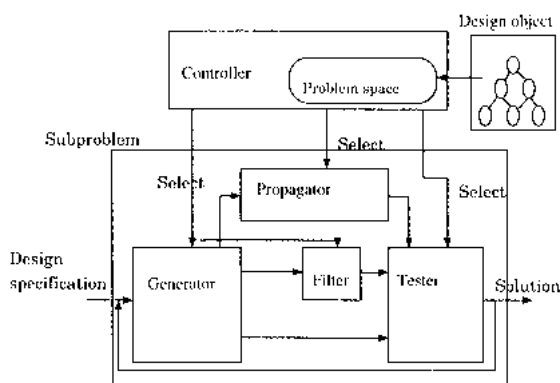


図2：ルーチン設計問題向け制約問題解決機構

緩和、代数的求解（連立方程式の求解）、推論の順序づけ、最適化などがおこなわれる。

一方、従来の問題解決では、問題領域と取り扱うべき対象を設定し、それらの関係を明確化し、さらに問題を解く手法を見出すことが必要であり、さらに、問題解決の制御に関する知識の過剰な記述や、冗長な記述、厳密な入出力関係を与えることが必要である。

制約問題解決のメリットとしては、処理の効率化と処理の高度化の2つの側面からなる問題解決能力の向上をあげることができる。まず、第1番目としては、局所化された整合性の保持手法、探索空間の大域的な（最適化）処理手法、ヒューリスティックスに基づいた探索などによる探索処理の効率化をあげることができる。第2番目としては、様々な適用領域向けの問題解決機構の提供による処理の高度（多機能）化をあげることができる。

次に、設計タスクに基づいた問題解決機構がどのような制約充足手法を用いて実現されるかについて簡単に説明する [33] [34] [35]。

たとえば、ルーチン設計問題を対象とした設計案の提示は、制約充足や最適化、あるいは代数解法を用いた制約問題解決により取り扱うことができる。ルーチン設計問題のなかで、設計構造が既知であり、設計パラメータを求めるようなパラメトリック設計では、設計案の提示のために数値的あるいは記号的制約を用いた制約充足手法や最適化手法が用いられる。また、このような制約充足処理では、制約がすべて満足されるような方向に制約伝播（記号制約および数値制約）がおこなわれることにより設計パラメータが決定され

る。このようにして、設計問題をすべて制約充足問題とみなすことはできるが、制約伝播や制約充足などの手法だけを用いて設計をした場合、それに要する計算が膨大なものとなり大規模な問題を取り扱うことは困難である。そこで、分解や事例による手法を用いて探索空間の規模を縮小し問題を小さな部分問題に分解できれば、大規模な探索をおこなわないで、制約充足を適用できる。これは、さきほど述べた探索をおこなう問題解決手法と解空間を探索せずに解を求めるアルゴリズムを用いた問題解決手法を統合した手法であると考えられる。

同様に、設計案の提示のなかで、分割-解合成法を用いた場合には、分割された問題の階層間では、パラメータや制約が階層の上位または下位に伝播されたり、制約により分割された部分問題間でのインタラクションが引き起こされたりする。その場合の制約の取り扱いには、最小拘束原理などの利用が有効である。設計案の評価では、制約の違反が設計にどのような影響を与えるかという情報を依存関係解析により求め、この情報が評価に用いられる。設計案の修正では、設計案の失敗に関する情報を用いて、設計要求を満足させるように設計を修正する。ここでは、失敗回復処理により違反制約に関する情報を用いることにより依存関係に基づいたバックトラック法により設計案の修正が行われる。

### 3.4 ルーチン設計問題向け知識システム実現に適用すべき制約指向技術の動向

以下では、ルーチン設計問題向け知識システム、特に問題解決機構の実現に有望な要素技術である制約指向技術について概観する。

#### ・知識コンパイル

設計問題向け知識システムが提供している知識表現や問題解決機構は従来の場合と比較するとより問題向けになっている。しかしながら、このようなシステムを利用してもエンドユーザによって構築される設計システムが、依然不十分であったり、非効率であることが多い。さらに、設計問題では様々な設計知識と問題解決機構が必要となるため、制約を有効利用して問題解決能力を向上させる手段として、知識変換技術を用いた知識の有効な活用法の一つで

ある知識コンパイル技術 [3] が必要とされる。制約に着目した知識コンパイル技術では、宣言的な知識表現である制約を手続き的な知識表現に変換し、問題解決機構が変換された知識を効率的に解釈・実行（解を導出）する。設計問題を対象とした場合には、設計タスク用の制約指向知識コンパイル技術が必要とされ、これより、宣言的に表現された設計知識から、あらかじめ設定された設計過程モデルに基づいて問題解決機構が効率よく実行可能な設計手順が生成される。さらに、機械のパラメトリック設計を対象とした設計支援システム構築ツール MECHANICOT では、このような制約指向知識コンパイル技術が適用されている [34] [35]。

#### ・制約管理

概念設計では、設計仕様が緩和、変更、修正されたり、制約が追加、削除、修正されることが多い。得られた制約集合には、矛盾が生じていたり、実現不可能な場合が考えられる。設計過程においては、これらの制約の管理をすることは容易ではない。制約管理では、このような制約の評価・実行ならびに整合性の維持がおこなわれる [43]。

#### ・動的制約充足

制約充足問題は変数集合とその要素である変数に対応する離散有限領域が固定されている場合に、制約を満足するように変数に対して領域から値を割り当てる問題である。この問題で利用される制約充足技術の設計問題への適用例として、自動車の変速機設計が知られている [31]。しかしながら、実際の設計問題では、問題解決時の推論に基づいて解に関係し値が割り当てられる変数集合が動的に変化する動的制約充足問題を解くことが必要である。このような動的制約充足はブリッジの設計 [22] や構成要素の配置問題 [29] などに適用されている。

#### ・制約充足最適化

事例ベースシステム CYCLOPS [37] では、景観設計を複数の目的関数を満足する配置問題とみなし、これを制約充足最適化により解いている。制約充足問題では最適化を考慮せずに列挙解を求めているのに対して、制約充足最適化では最適化を考慮しているところに特徴がある。このような最適化をとまなう制約充足問題として設計問題を定式化する場合には、制約充足最適化手法が必要とされる。

#### ・制約指向言語

##### －制約論理プログラミング言語の適用

論理型言語の宣言的な側面と制約問題解決（特に、制約充足）における効率化手法を組み合わせた制約論理プログラミング言語 CHIP [11] は、実際的なアプリケーションを制約を用いた探索問題として解くことを目的とした言語である。この言語では、制約を能動的に用いて探索空間を縮小させる探索手法が用いられている。その応用として、スケジューリングや設計問題が取り上げられており、特に、大規模な回路の設計（回路シミュレーション、機能仕様の検証、組み合わせ回路の生成、故障診断、テスト・パターン生成、マイクロコードにおけるラベルの割り当てなど）に有効であることが報告されている。設計問題を含む合成型問題に対する、CHIP に代表される制約論理プログラミング言語の適用は有望であると思われる。

##### －階層制約論理プログラミング言語の適用

ユーザ・インターフェイス設計やレイアウトなどの合成型問題では、制約としてかならず満足しなければならない制約、すなわち必須制約だけでなく、「…したほうがよい」といった示唆といった概念が記述できる示唆的な制約を必要とする場合が少なくない。また、このような問題では複数の可能解から最適解を決定するために必須制約より弱い意味付けのなされた目的制約が必要となる場合も考えられる。したがって、このような示唆的な制約や目的制約からなる制約階層が取り扱える階層制約論理プログラミング言語 [6] の利用により、示唆的な制約や目的制約を、システムの手続き部分に直接コード化するのでなく、宣言的に制約として記述できるので、制約階層を取り扱った設計問題向け知識システムの構築が容易化することが期待できる。

## 4. おわりに

本論文では、設計問題向け知識システムにおける制約指向技術の適用について、ルーチン設計問題を対象にして検討した。まず、設計システムについて要求される一般的な機能について考察した。そして、設計過程モデルを明確にするために、設計対象についての分類を行い、基本となる設計タスクについて述べた。次

に、上記のなかで特に、様式化の図られている設計過程により問題自身をwell-definedかつwell-structuredな問題として取り扱うことが可能である設計問題をルーチン設計問題と定義した。そして、より詳細なルーチン設計の分類を行い、実際の設計対象領域と対応付けた。最後に、ルーチン設計問題を対象とした知識システムに要求される機能の実現に必要なとなる制約指向技術とその適用について説明した。

今後は、機械設計以外のルーチン設計問題（たとえば、LSI設計、ハードウェアコンパイラなど）についての検討にも力をいれていく予定である。最終的には、様々な設計対象に対する設計問題についての分析を行い、設計問題向け知識システムのアーキテクチャの分類化（共通部分と異なる部分との明確化）とその際に必要となる要素技術（特に、制約指向技術）について明確化を図っていきたい。

## 参考文献

- [1] Aho, A.V. and Ullman, J.D., Principles of Compiler Design, Addison-Wesley Publishing Company Inc., (1977).
- [2] 相場亮, 古川康一, 制約プログラミングについて－制約ロジックプログラミングを中心として－, 人工知能学会誌, Vol.6, No.1, pp.47-59, (1991).
- [3] Araya, A. and Mittal, S., Compiling Design Plans from Descriptions of Artifacts and Problem Heuristics, IJCAI 87, Vol.2, pp.552-558, (1987).
- [4] Begg, V., Developing Expert CAD Systems, New Technology Modular Series, Kogan Page, (1984).
- [5] Bermingham, W. and Siewiorek, D., MICON: A Knowledge Based Single Board Computer Designer, IEEE 21st Design Automation Conference, pp.565-571, (1984).
- [6] Borning, A. *et.al*, Constraint Hierarchies, Proc. OOPSLA '87, pp.48-60, (1987).
- [7] Brewer, F. and Gajski, D., An expert-system paradigm for design, IEEE 23rd Design Automation Conference, (1986).
- [8] Brown, D.C. and Chandrasekaran, B., Knowledge and Control for a Mechanical Design Expert System, IEEE COMPUTER, (1986).
- [9] Chandrasekaran, B., Generic Tasks in Knowledge-based Reasoning: High-Level Building Blocks for Expert System Design, IEEE expert, FALL, (1986).
- [10] Chandrasekaran, B., Design Problem Solving: A Task Analysis, AI Magazine, Vol.11, No.4, pp.59-71, Winter, (1990).
- [11] Dincbas, M. *et.al*, The Constraint Logic Programming Language CHIP, Proc. Int. Conf. on Fifth Generation Computer Systems, (1988).
- [12] Clowes, M.B., On seeing things, Artificial Intelligence 2, (1971).
- [13] Davis, E., Constraint Propagation with Interval Labels, Artificial Intelligence 32, (1987).
- [14] Dechter, R., Network-Based Heuristics for Constraint-Satisfaction Problems, Artificial Intelligence 34, pp.1-38, (1988).
- [15] Dincbas, M., CONSTRAINTS, LOGIC PROGRAMMING AND DEDUCTIVE DATABASES, France-Japan Artificial Intelligence and Computer Symposium 86, (1986).
- [16] Dixon, J.R. and Simmons, M.K., Expert systems for Design: Standard V-Belt Drive Design as an Example of the Design-Evaluate-redesign Architecture, Procs ASME Computers in Engineering Conference, Boston, MA, August, (1984).
- [17] Dixon, J.R., Simmons, M.K. and Cohen, P. R., An Architecture for applying Artificial Intelligence to Design, Procs IEEE DAC, Albuquerque, NM, June, (1984).
- [18] Doyle, J., A Truth Maintenance System, Artificial Intelligence 12, (1979).
- [19] Eastman, C., Recent Developments in Representation in the Science of Design, IEEE 18th Design Automation Conference, pp.13-21, (1981).
- [20] Freuder, E.C., A sufficient condition of backtrack-free search, Journal of ACM 29 (1), (1982).
- [21] Fox, M.S., Constraint-Directed Search: A Case Study of Job-Shop Scheduling, CMU-RI-TR-83-22, (1983).
- [22] K. Hua *et.al*, Dynamic Constraint Satisfaction in a Bridge Design System, Lecture Notes in Artificial Intelligence 462, Expert Systems in Engineering, (ed. G. Gottlob and W. Nejdl), pp.217-232 (1990).
- [23] Jaffar, J. and Lassez, J.-C., Constraint Logic Programming, (1986).
- [24] Johannsen, D., Bristle Blocks, A Silicon Compiler, IEEE 16th Design Automation Conference, (1979).
- [25] Kowalski, T. and Thomas, D., The VLSI Design Automation Assistant: Prototype System, IEEE 20th Design Automation Conference, pp.479-483, (1983).
- [26] McDermott, J., Domain Knowledge and the Design Process, IEEE 18th Design Automation Conference, pp.580-588, (1981).
- [27] Medland, A.J., The computer-based design process. 1. Engineering design - Data processing I, Kogan Page Ltd, (1986).
- [28] Mittal, S. and Araya, A., Knowledge-Based Framework for Design, Proc. of AAAI-86, pp.856-865, (1986).

- [29] Mittal, S. and Falkenhainer, B., Dynamic Constraint Satisfaction Problems, Proc. of IJCAI89, pp.25-32, (1989).
- [30] Murthy, S. and Addanki, S. PROMPT: An Innovative Design Tool, AAAI '87, (1987).
- [31] Nadel, B. and Lin, J., Automobile Transmission Design as a Constraint Satisfaction Problem: Modelling the Kinematic Level, AI EDAM, Vol.5, No.3, pp.137-171, (1991).
- [32] 永井保夫, 設計型エキスパートシステム研究に関するイメージ及び技術課題について, 昭和62年度KSS-WG DES報告書, ICOTテクニカルメモランダムTM-681, (1989).
- [33] 永井保夫, 設計問題向けツールアーキテクチャ, 人工知能学会誌, Vol.4, No.3, pp.297-303, (1989).
- [34] 永井保夫他, 設計問題向け制約指向知識コンパイラにおける制約解析および手順生成について, 人工知能学会第3回全国大会, 11-43, pp.693-696, (1989).
- [35] Nagai, Y. and Terasaki, S., A Constraint-Based Knowledge Compiler for Parametric Design Problems in Mechanical Engineering, 人工知能学会誌, Vol.11, No.1, pp.60-74, (1997).
- [36] 長澤勲, 設計エキスパートシステム, 情報処理学会誌, Vol.28, No.2, (1987).
- [37] Navinchandra, D., Exploration and Innovation in Design: Towards a Computational Model, Springer-Verlag (1991).
- [38] Nicklaus, D.J., Tong, S.S. and Russo, C.J., ENGINEOUS: A KNOWLEDGE DIRECTED COMPUTER AIDED DESIGN SHELL, The Third Conference on Artificial Intelligence Applications, February, (1987).
- [39] Nudel, B., Consistent-Labeling Problems and their Algorithms: Expected-Complexities and Theory-Based Heuristics, Artificial Intelligence 21, (1983).
- [40] Rao Palada, A.M. and Bose, P.K., ASSUMPTION BASED REASONING APPLIED TO PERSONAL FLIGHT PLANNING, AVIGNON 87, (1987).
- [41] 白井克彦, 永井保夫, 竹沢寿幸, 高級言語による仕様記述に基づく回路のデータベース系自動設計法, 情報処理学会論文誌, Vol.28, No.1, pp.99-108, (1987).
- [42] Southard, J.R., MacPitts: An Approach to Silicon Compilation, IEEE Computer, (1983).
- [43] Sriram, D. et.al, Knowledge-Based System Applications in Engineering Design: Research at MIT, AI Magazine, Vol.10, No.3, pp.79-96, Fall (1989).
- [44] Stallman, R.M. and Sussman, G.J., Forward Reasoning and Dependency-Directed Backtracking in a System for Computer-Aided Circuit Analysis, Artificial Intelligence 9, (1977).
- [45] Subrahmanyam, P.A., Synapse: An Expert System for VLSI Design, IEEE Computer, July, (1986).
- [46] Sussman, G.J. and Steel, G.L. Jr., CONSTRAINTS—A Language for Expressing Almost-Hierarchical Descriptions, Artificial Intelligence, Vol. 14, (1980).
- [47] Sutherland, I.E., SKETCHPAD: A Man-Machine Graphical Communication System, Proc. of Spring Joint Computer Conference, (1963).
- [48] Tomiyama, T. and Hagen, P.J.W.T., Organization of Design Knowledge in an Intelligent CAD Environment, in Expert Systems in Computer-Aided Design (Gero (ed.)), pp.119-152, North-Holland, (1987).
- [49] 富山哲夫, 吉川弘之, 設計過程モデル論, 精密機械49巻4号, (1983).
- [50] Tong, C., Toward an Engineering Science of Knowledge-Based Design, Artificial Intelligence in Engineering, Vol.2, No.3, pp.133-166 (1987).
- [51] Waltz, D.L., Generating semantics descriptions from drawings of scenes with shadows, AI-TR-271, AI-Lab., MIT, (1972).
- [52] Wolf, W.H., Kowalski, T.J. and McFarland, S.J., Knowledge Engineering Issues in VLSI Synthesis, AAAI '86, (1986).
- [53] 吉川弘之, 機械のトポロジ, 精密機械38巻12号, (1972).
- [54] 吉川弘之, 一般設計過程, 精密機械47巻4号, (1981).
- [55] 吉川弘之, 設計とは何か 一般設計学の試み, 日本機械学会誌 8巻749号, (1981).