知識工学と人工知能 31－7
（1983. 9. 27）

プロローグを基礎とした設計システム記述言語ＡＤＬ

長澤　勲＊　古川　由美子＊　荒牧　重登＊＊

＊九州大学中央計数施設　　＊＊九州大学情報処理教育センター

１．まえがき

設計作業とは，一般に構成，解析，評価，最適化といった過程を経て，性能，信頼性，使い易さ，経済性，加工性などの項目の調和をはかって行くものである[1,2]．現在までに開発されてきた設計支援システムの多くは，これらの過程のうち主として解析過程の支援を目的としている．この理由は，解析過程では汎用解析モデルを用いることができ，自動化し易く，効果も大きいからであると考えられる．これに対して構成過程では，所要機能・性能に対して必要な構造・構造諸元を決定するのが主題であり，また考慮しなければならない設計要件も様々である．このため，構成過程の支援は解析過程ほど統一的に行えないという難点がある．しかし構成作業においても機械化するのに大きい意味のある部分がある．たとえば，設計資料を調べ部品の強度計算をし，使用可能な部分をカタログからさがしたり，あらかじめわかっている構成方法の中から適当なものを選択することが考えられる．

そこで，筆者らは構成過程のうちこれらの作業を支援するシステムＡＤＬ（Ａ　Ｄｅｓｉｇｎｅｒ’ｓ　Ｌａｎｇｕａｇｅ）を開発した．本論文ではこのシステムの基本概念，実現法ならびに設計知識の表現法について述べるものである．構成過程を支援するシステムの試みは，問題解決法を用いたもの[10]と拘束条件解法を基礎にしたもの[2,11,12,13]が提案されている．前者は構成過程のうち主として所要機能・性能から構造を決定する構造設計に適しており，複雑な機能をもつ対象の設計には不可欠な手法である．しかし一般にこのレベルの設計知識は現状では十分整理されておらず実用化は容易ではない．後者は，機能と構造，所要性能と構造諸元等の拘束関係を利用するものであり，構造諸元の決定やパターン化された構造の選択に適している．また，これに必要な設計知識は，すでに設計資料，設計公式，設計規約などの形で十分整理されている[4]．ここでは後者の方法をとることにする．拘束条件知識を基礎にして，設計支援システムを構成するには，設計公式の数値・数式処理，設計資料検索，生成検証法，伝播法を始めとする解探索法，利用者との柔軟な対話環境などを実現する必要がある．現在までに提案されている拘束条件を基礎としたシステム[2,11,12,13]はこれらの機能を個別に取り扱っており，統一的方法で解決しているとは言い難い．一方，論理プログラミングは，データベース検索[14]，数式処理[15]，対象指向プログラミング[5,6]などへの適応性が高いことが知られており，ここでの目的に最適であると考えられる．筆者らは，ｐｒｏｌｏｇを核言語にとり，設計支援に必要な種々の機能を実現することにした．

２．ＡＤＬシステムの概要

ＡＤＬシステムは，図１に示すように２つのモジュールＫＢＥ（Ｋｏｗｌｅｄｇｅ　ｂａｓｅ　ｅｄｉｔｏｒ）とＣＲＳ（Ｃｏｎｓｔｒａｉｎｔｓ　ｒｅｄｕｃｔｉｏｎ　ｓｙｓｔｅｍ）から構成される．設計知識ベースには，設計要求，設計結果および設計知識が記憶さ

表1　設計知識と主な用途

Table 1　Design knowledges and their use

方法	主に適応する設計過程	知識の完成度・性質	例
問題解決法	構造設計	不完全・手続的	構成法の発見規則
拘束条件解法	構造諸元決定・構造設計	完全・宣言的	設計公式・資料・規格
解析モデル計算	解析	完全・宣言的	有限要素法・回路解析

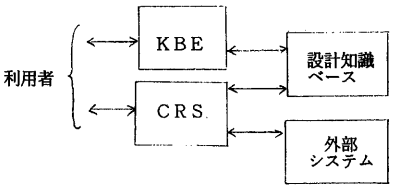

図1　ADLの構成

Fig. 1 An overview of ADL system

れ，ＫＢＥはこれらの編集・管理に使用される．ＣＲＳはｐｒｏｌｏｇに拘束条件リダクションおよび対象指向プログラミング手法を拡張したものであり，設計計算，設計資料検索，設計の検証などに使用する．ＣＲＳはこのほか数値計算，図形処理などを行うため外部システムとのインターフェイスを持っている．以下本稿では，ＫＢＥの詳細は省略し，主としてＣＲＳについて述べる．

3．　拘束条件リダクションシステム

設計問題は拘束条件集合の形で与えられる．ＣＲＳはこれをリダクションによって解くシステムである．ＣＲＳは拘束条件解法のうち，生成検証法，拘束条件伝播法[11,12]などの実現を意図しており，自動バックトラック，データフロー制御，コスト評価による効率改善等の機能を持っている．

2．1　記述形式

ＣＲＳはｐｒｏｌｏｇの手続の集まりとして実現し，特別の記述形式を用いない．

［拘束条件］拘束条件は素論理式で表現する．この論理式は，ｐｒｏｌｏｇのｕｎｉｔ　ｇｏａｌとして使用することが多いが他の用法もあるので以下〃拘束条件〃とよぶ．

［ＣＲＳの呼び出し］ＣＲＳの呼び出しにはｓｏｌｖｅ述語を用い，次のように記述する＊．

：－（ｓｏｌｖｅ　Ｃ１，Ｃ２，…，Ｃｎ）．

Ｃ１，Ｃ２，…，Ｃｎは拘束条件である．これらは，記述順序に意味を持たず集合として取り扱われる．

［リダクション・ルール］リダクション・ルールはｐｒｏｌｏｇの手続で表現する．リダクション・ルールを構成するｃｌａｕｓｅの一般形式は次で与えられる．

Ａ：－Ｇ１，…，Ｇｉ，！！，Ｇｉ＋１，…，Ｇｎ．

または，Ａ：－Ｇ１，…，Ｇｎ．（ｎ≧０）

Ｇ１，…，Ｇｎはｕｎｉｔ　ｇｏａｌである．Ｇ１，…，Ｇｎの評価中に別のリダクション・ルールを呼び出してはならない．リダクション・ルールとしての意味を補強するため２つの基本述語！！，ｄｅｖａｌを導入する．！！はカット・オペレータの一種であり，リダクション・ルールの中にだけ書くことができる．！！は通常のカットの作用以外に，後述するように拘束条件の評価の意味を修飾する．ｄｅｖａｌ述語は，Ｇ１，…，Ｇｎを評価中に呼

び出すことができ，リダクションによって置換される拘束条件集合をＣＲＳに通知するのに使用される．ｄｅｖａｌ述語の形式は（ｄｅｖａｌ　Ｄ１，Ｄ２，…，Ｄｎ）である．Ｄ１，…，Ｄｎは拘束条件である．

［変数修飾子］データフローを表現するために論理変数に修飾子を付加する．変数に前置された？を入力修飾子，変数に後置された？を出力修飾子とよぶ＊＊．修飾子は変数の一部ではなく，たとえばＸを変数とするとき，？Ｘ，Ｘ？，Ｘは同一変数である．また，修飾子は変数の値が変数である限り遺伝する＊＊＊．ｕｎｉｆｉｃａｔｉｏｎにおける変数の代入規則を表２に示す．

表２　修飾付変数の代入規則

Table 2　Substitution rule of annotated variables

ゴール側 / 手続側	非変数	Y?	?Y	Y
非変数	＊	×	↑	↑
X?	×	×	×	×
?X	←	×	×	×
X	←	×	×	←

× 失敗　　X?, Y?　出力修飾付変数
← 手続側変数への代入　　?X, ?Y　入力修飾付変数
↑ ゴール側変数への代入　　X, Y　通常の変数
＊ 通常の項の統一化

ステップ0　　Ｓｉ＝Ｓ０（初期拘束条件集合）とおく。

ステップ1　　次の３つの場合がある．
（ｉ）拘束条件集合が空集合であれば停止する．
（ii）未評価な拘束条件がなければデッドロック解消を試みる．成功すればステップ１へ，失敗すれば停止する．
（iii）拘束条件集合　Ｓｉ＝｛Ｃ１，Ｃ２，…，Ｃｍ｝　から任意に未評価の拘束条件Ｃｊを選択する。このときバックトラックにそなえて未評価な拘束条件集合を記録しておく．ステップ２へいく．

ステップ2　　Ｃｊのリダクションを試みる。次の３つの場合がある。
（ｉ）Ｃｊの評価結果がｓｕｓｐｅｎｄならばステップ１へバックトラックする。
（ii）Ｃｊの評価結果がｆａｉｌならば前サイクルのステップ２（iii）へバックトラックする。
（iii）Ｃｊの評価結果がｓｕｃｃｅｓｓであり，Ｃｊの評価中に（ｄｅｖａｌ　Ｄ１，…Ｄｎ）を実行したとすると次のサイクルの拘束条件集合は
Ｓｊ＋１＝｛Ｃ１，Ｃ２，…，Ｃｊ－１，Ｄ１，Ｄ２，…Ｄｎ，Ｃｊ＋１，…Ｃｍ｝θ
である。θはＣｊの評価中に行った代入である。また，リダクションの仕方が一意でない場合バックトラックへそなえて別解を記録する。ステップ１へ行く．

図２　リダクション手続

Fig. 2 Reduction procedure of ADL／CRS.

２．２　リダクション手続

まず，一つの拘束条件を評価する手順について述べる．今，リダクションを試みる拘束条件をＣとする．

ＣはＣＲＳによってｐｒｏｌｏｇのｕｎｉｔ　ｇｏａｌとみなして評価されるが，評価結果は次の３つに分類される．

(1)ｆａｉｌ　Ｃの評価がｐｒｏｌｏｇの意味で失敗であるが！！が実行されているとき．この場合積極的な否定情報を持つとみなす．

(2)ｓｕｃｃｅｓｓ　Ｃの評価に成功し，その評価中に（ｄｅｖａｌ　Ｄ１，…，Ｄｎ）を評価するとＣは｛Ｄ１，…，Ｄｎ｝θにリダクションされる．ただしθはＣの評価中に行っ

＊ｌｉｓｐのＳ式で表わす．また，英大文字で始まる文字列は変数，英小文字で始まる文字列は定数である．

＊＊修飾子の入力・出力はｕｎｉｆｉｃａｔｉｏｎにおける変数の代入方向を表わしている．

＊＊＊Ｃｏｎｃｕｒｒｅｎｔ　ｐｒｏｌｏｇ　を参考にしている．

た代入，{} は集合を表わす．ｄｅｖａｌを評価していなければＣは空集合にリダクションされる．

⑶ｓｕｓｐｅｎｄ　上記以外の場合，即ちｐｒｏｌｏｇの意味でｆａｉｌであり！！を実行していないとき．この場合何ら積極的情報を持たないとみなす．

［リダクション過程］リダクションの各段階における拘束条件集合をＳｉで表わすと，リダクションにおける拘束条件集合の系列は，Ｓ０，Ｓ１，Ｓ２，…，Ｓｉ，…，Ｓｎとかける．Ｓ０は初期拘束条件集合，各ＳｉはＳｉ－１からリダクションによって得られた拘束条件集合である．評価してもｓｕｃｃｅｓｓまたはｆａｉｌする拘束条件がなくなったときリダクションは停止する．リダクション手続を図２に示す．

［デッドロックの解消］拘束条件集合は個別にリダクションする方法では解けないことがある．しかし，連立方程式のように専用の解法を用いれば解ける場合，システムの使用者の介入によって拘束条件を追加すれば解ける場合などがある．ＣＲＳはシステムの標準として連立方程式の解法を用意しているが利用者の目的に応じて機能を追加できる．

［解法の階層化］拘束条件集合のリダクション順序はデータフローを満足すれば任意でよい．しかし，設計計算には，自然な階層が存在することが多い．これを表現するためリダクション・ルールに優先順位を与え，優先順位の高い拘束条件のリダクションを優先する．

４．　構造物の表現

図３は，平歯車システムとその設計に必要な拘束条件である．比較的簡単な構造物でも多くの部分や属性を持つことがわかる．フレーム表現[17]は，このような構造物を表現する手段として適しており，次のような特徴があることが知られている．⑴対象指向な知識表現の自

減速比	u	
ピッチ円直径	d	[㎜]
モジュール	m	[㎜]
ピッチ円周速度	v	[m／ｓｅｃ]
歯数	Z	[枚]
回転数	n	[ｒｐｍ]
伝達動力	L	[Ｋｗ]
伝達トルク	T	[kgｆ・㎜]
曲げ応力	σ	[kgｆ／㎜]
繰り返し許容曲げ応力	σ	[kgｆ／㎜]
歯幅	b	[㎜]
歯形係数	Ｋｚ	
速度係数	Ｋｖ	
衝撃係数	K	

$u = Z / Z$

$u = n / n$

$d = m \cdot Z$

$b = 10 \cdot m$

$v = \pi \cdot d \cdot n / (60 \times 10^{3})$

$T = 9.74 \times 10^{5} \times L / n$

$\sigma = 2 \cdot K \cdot T / (m \cdot Kv \cdot b \cdot Kz \cdot Z)$

$\sigma < \sigma$

平歯車の設計には、歯の面圧強さ，曲げ強さ，歯面仕上げ，効率等を考慮しなければならないが，ここでは，問題を簡単にするため曲げ強さだけを考慮している。歯の曲げ強さを求める公式はルイスの式を用いている．

図３　平歯車システムの拘束条件

Ｆｉｇ．３　Ｃｏｎｓｔｒａｉｎｔｓ　ｏｆ　ａ　ｓｐｕｒ　ｇｅａｒ　ｓｙｓｔｅｍ．

```
<フレーム>　::=
  (ｆｒａｍｅ<フレーム名>
     (Ｏｂｊ［(ａｋｏ　<フレーム名>‥‥<フレーム名>)］
           <スロット>‥‥<スロット>)
     (ｐｒｏｃ　<拘束条件>‥‥<拘束条件>)))
<フレーム名>::=　<文字定数>
<スロット>　::=(<スロット名><変数>)
<スロット名>::=　<文字定数>
<フレーム参照>::=
  (ｆｇｅｔ<オブジェクト参照>‥‥<オブジェクト参照>)
<オブジェクト参照>::=
  (<識別子>［(ａｋｏ　<フレーム名>)］
    <スロット参照>‥‥<スロット参照>)
<識別子>　　::=<文字定数>｜<変数>
<スロット参照>::=(<スロット名><ｐｒｏｌｏｇの項>)
```

図４　フレーム定義と参照の形式

Ｆｉｇ．４　Ｓｙｎｔａｘ　ｏｆ　ｆｒａｍｅ　ｄｅｆｉｎｉｔｉｏｎ　ａｎｄ　ｒｅｆｅｒｒｅｎｃｅ．

然な単位である．(2)手続附加により宣言的知識表現と，手続的知識表現が融合できる．(3)遺伝階層の利用により，知識表現がモジュール化できる．

機械や電子回路の設計分野では，対象指向に設計法や設計公式が開発されており，これらの知識をフレームを用いて表現するのは自然である．ここでは，拘束条件を伴った対象のクラスを表現するためフレームを用い，手続附加を前節で導入した拘束条件の集合でおきかえる．図4にフレームの記述形式を示す．スロットは対象の属性や，部分を示すのに使用しスロット名，変数の対からなる．スロットと拘束条件は変数を共有する．ａｋｏスロットは上位フレームを参照する．個々の対象物の表現をフレーム・インスタンスとよぶ．フレーム・インスタンスの生成やその間の結合には，ｆｇｅｔ述語を使用する．

［ｆｇｅｔ手続の概要］　システムには生成されたフレーム・インスタンスを管理する対象リスト（ｏｂｌｉｓｔ）と，リダクション途中の拘束条件集合を管理する拘束条件リスト（ｃｌｉｓｔ）の2つのデータ構造がある．ｆｇｅｔは次の作業を行う．(1)フレーム参照の中で指定した識別子に対応するフレーム・インスタンスがｏｂｌｉｓｔになければ新たにフレーム・インスタンスを生成しｏｂｌｉｓｔにのせる．またフレームの拘束条件部をｃｌｉｓｔに併合する．(2)フレーム参照とフレーム・インスタンスのスロット値を，ｕｎｉｆｙする．

5．　ＣＲＳの表現能力

本章では，設計計算の範例を取り上げ，ＣＲＳの表現能力を検証する．

5．1　設計プログラムの例

文献[10]参照．

5．2　ＣＲＳにおけるプログラム手法の例

［生成・検証法］文献[10]参照．

［状態仮説法］状態仮説法は電子回路の素子モデル，材料力学の部材モデルなどに使用される．図5(a)は抵抗器と理想ダイオードの直列接続に１０ボルトの電圧を加えたものである．

［伝播法］Ｓｕｓｓｍａｎらは線形連立方程式を解く一方法[11,12]を提案している．次の問題は数値を伝播させて解くことはできないが連立方程式の変数消去法を用いれば容易に解くことができる（図5(b)参照）．

：－（ｓｏｌｖｅ　（Ｘ＋Ｙ＝３）（Ｙ＝２＊Ｘ））

これには方程式を解くリダクション・ルールに次の機能があればよい．(1)変数がなければ等式を検証する．(2)一変数方程式は無条件に解く．(3)多変数方程式は一変数について解く．

［非線形連立方程式解法］野口らは設計計算に必要な非線形方程式の解法として数式を単項演算子式に分解し，しかる後，数値計算を行う方法[7]を提案している．この方法は数式処理と数値計算を組み合わせたものである．前半の処理は図5(c)のリダクション・ルールによって，後半は図6のデッドロック解消手続ｍｏｎｉｔｏｒから数値計算ライブラリを呼び出す

ことによって容易に実現できる.

```
:- (solve
    (10=Vr+Vd) (register Vr I 10)
    (diode Vd I State)).
(diode Vd 0 off) :- (deval (Vd <= 0)).
(diode 0 Id on) :- (deval (Id >= 0)).
(register Vr Ir R) :- (Vr=Ir*R).
```

(a) 状態仮説法の一例

```
(X=Y) :- (free_of_variable (X=Y)),!!,
         (approx_equal X Y).
(X=Y) :- (single_variable (X=Y)),!,
         (solve_equation (X=Y)).
(X=Y) :- (select_variable (X=Y) Z),
         (solve_equation_with (X=Y) Z).
```

(b) 拘束条件伝播法

```
(X=Y) :- (free_of_variable (X=Y)),!!,
         (approx_equal X Y).
(X=Y) :- (single_variable (X=Y)),!,
         (solve_equation (X=Y)).
(X=Y) :- (multiple_operator (X=Y)),
         (decompose_equ (X=Y) Eq1 Eq2),
         (deval Eq1 Eq2).
```

(c) 一演算子式への分解

図5 プログラムの例題

Fig. 5 Programming examples

6. ＣＲＳの実現と最適化

ＡＤＬは，ＦＡＣＯＭ　ＯＳⅣ／Ｆ４　ＬＩＳＰ[8]上に実現されている．システムの詳細は別稿[9]に譲り，ここではＣＲＳの実現と効率改善の一方法について述べる．

6．1　制御方法

図６にＣＲＳの主要部を示す．ＣＲＳは，４つのデータ構造を持っている．Ｒｅａｄｙ－Ｒｔ，Ｗａｉｔ－Ｗｔはそれぞれ，未評価な拘束条件集合，評価を行ったがｓｕｓｐｅｎｄされた拘束条件集合を表わし，それぞれｄ＿ｌｉｓｔ＊により実現している．ｃｌｉｓｔ，ｏｂｌｉｓｔはＣＲＳとリダクション・ルール間の交信に使用される広域的データ構造であり，拘束条件集合，フレーム・インスタンス集合を表わしている．Ｒｅａｄｙ－Ｒｔ，Ｗａｉｔ－Ｗｔ，ｃｌｉｓｔは効率改善のため後述する優先度，コストによってソートされている．

＊リストをポインターの差で表現する方法

6．2　コスト評価による効率改善法

拘束条件集合の評価は，データフローの制限を満足しさえすればどのような順序で行っても同じ結果を得ることができる．しかし，生成検証法のようにバックトラックを多用する応用では，効率に大巾な影響を与える．ここでは拘束条件に優先順位，動的コストを定義し，これを用いて拘束条件の評価順序を決定する．

各拘束条件に優先順位ｐ，動的コストｃ＝ｎ・ｓを定義する．ｐ，ｓはそれぞれリダクション・ルールに定義した優先順位およびコストである．ｐは，小さい程高順位である．また，ｓはルール評価に必要な計算量，非決定性の程度によって決定しておく．ｎは拘束条件に出現する変数の個数であり，リダクションの進行とともに動的に変化する．ＣＲＳ起動時および，ｄｅｖａｌ手続き中では，拘束条件集合を優先順位ｐ，コストｃの小さい順にソートする．リダクションの各段階では最小コストの拘束条件を求めている．

コスト制御の効果を調べるため歯車減速機の基本設計を行うプログラムを用い簡単な実験を行った．表３においてリダクション・ルールのコストｓは同一にしてある．オーバーヘッドは総計算量に対する制御に要した計算量をｕｎｉｆｉｃａｔｉｏｎ回数で測定した結果，(a)１６～２５％ (b)１．６～３％ (c)１．０～２．７％であった．またリダクション・ルールのコストｓに対する影響を調べるため，ｓをリダクションがｓｕｓｐｅｎｄされたとき１加算，ｓｕｃｅｓｓまたは，ｆａｉｌのとき１減算して調整したものを準備し同様に測定した．(a)ではｓｕｓｐｅｎｄ回数は大巾に減少するが計算時間は，ほとんど変化がなく，(b)ではｓｕｓｐｅｎｄ回数，計算時間共改善されるものとされないものがあって多少不安定であった．以上の結果により，コスト評価による効率改善法はある程度効果があるといえる．

７．　あとがき

本論文では，設計システム記述言語ＡＤＬを提案し，基本概念，実現法，表現能力について述べた．現在，筆者らは，ＡＤＬを用いて機械設計の教育を目的として，歯車減速機の設計システムを開発している．この開発を通して，設計計算を統一的方法で表現し，可用性，拡張性，保守性の高い設計システムの構成法を考案するという初期の目的はひとまず達成できたことを確認している．しかし，より実際的な，設計システムを実現するにはより一層の機能拡張が必要である．機能拡張の一つとしてＴＡＬＫと呼ぶ対象指向プログラミングの手法を用いた会話型モジュールの開発を始めている．また，教育用応用システムとして材料力学のコンサルテーションシステムの開発を行う予定である．

謝辞　本論文をまとめるにあたり，文献を提供され，貴重な助言をいただいたＩＣＯＴ第二研究室古川康一室長を始め同研究室の諸氏，ならびに日頃助言をいただく，九州大学工学部吉田将教授，牛島和夫教授，中央計数施設大槻説乎助教授，大型計算機センタ松尾文碩講師に謝意を表わす．

【参考文献】

１）和久井，下村：電子回路のＣＡＤ，ｐ．２３６，日刊工業新聞社（１９７２）．
２）沖野　教郎：自動設計の方法論，ｐ．１９２，養賢堂（１９８２）．
３）小川　潔：　機械設計システムのプログラミング，ｐ．２０６，森北出版（１９７７）．
４）小川　潔：　機械設計システム，ｐ．２４０，森北出版（１９７３）．
５）竹内，古川，Ｓｈａｐｉｒｏ，Ｅ．Ｙ．：Ｃｏｎｃｕｒｒｅｎｔ　ｐｒｏｌｏｇによるオブジェクト指向プログラミング，ロジックプログラミングコンファレンス（１９８３）．
６）服部　隆：オブジェクト指向的なｐｒｏｌｏｇプログラミング，ロジックプログラミングコンファレンス（１９８３）．
７）野口，安藤：非線形連立方程式の自動求根プログラム，情報処理学会論文誌，Ｖｏｌ．２４，ｐｐ．１３７－１４２（１９８３）．
８）計算機マニュアル　ＦＡＣＯＭ　ＯＳⅥ／Ｆ４　ＬＩＳＰ手引き書．
９）長澤，古川：ｐｒｏｌｏｇ系言語　ＡＤＬ（１）－　ＡＤＬの概要とインタプリター，九州大学大型計算機センタ広報，Ｖｏｌ．１６，ｐｐ．２５２－２７９　（１９８３）．

```
solve(Constraints) :-
      append(Constraints,Rt,Ready),
      quick_sort(Ready-Rt,Sorted-St),
      set(clist,Rt-Rt),reduce(Sorted-St,Wt-Wt).
reduce(Rt-Rt,Wait-Wt) :-
      var(Rt),!,monitor(Wait-Wt).
reduce([R|Ready]-Rt,Wait-Ready) :-
      set_onbacktrack(clist,Wait-Rt),
      off_flag,R,!&,value(clist, Rdy1-Rt1),
      find_mincost_constraint(Rdy1-Rt1,Rdy2-Rt1),
      reduce(Rdy2-Rt1,Wt-Wt).
reduce([R|Ready]-Rt,Wait-[R|Wt]) :-
      suspendp,reduce(Ready-Rt,Wait-Wt).
monitor(Wait-Wt) :-
      pick_equations(Wait-Wt,Equs,Ready-Rt),
      solve_equations(Equs),!,
      reduce(Ready-Rt,Wt-Wt).
monitor(Wait-Wt) :-
      value(oblist,Obl),fassert(Obl).
deval(Constraints) :-
      append(Constraints,Rt,Ready),
      deval1(Ready-Rt).
fget(Objs) :-
      match(Objs,Constraints-Rt),
      deval1(Constraints-Rt).
deval1(Constraints-Ready) :-
      value(clist,Ready-Rt),
      quick_sort(Constraints-Rt,Sorted-St),
      set_onbacktrack(clist,Sorted-St).
```

図6 CRSインタプリタの概要

Fig 6 Outline of CRS interpreter.

＊ set_onbacktrackは第2引数を第1引数に設定する副作用である。またリダクションの失敗によってバックトラックしたとき第1引数の値を復元できる。

＊＊ off-flagはシステムフラグを初期設定する。リダクション・ルール実行中！！が評価されるとこのフラグが反転することを利用してfailとsuspendの区別をしている。

＊＊＊ !&はカットオペレータの一種であり，親レベルのバックトラック点だけを除去する。

表3 コスト制御による効率改善の実測

Table 3 Results of experiment using the cost controll scheduling.

	prob.	call	exit	fail	susp	cpu(ms)
	(1)	39	27	8	12	955
(a)	(2)	41	15	0	26	442
	(3)	59	51	32	8	1927
	(1)	146	35	16	111	1101
(b)	(2)	44	15	0	29	314
	(3)	323	91	72	232	3078
	(1)	157	88	69	69	2538
(c)	(2)	50	15	0	35	217
	(3)	294	151	132	143	3481

call：リダクション・ルールの呼び出し回数
exit：リダクションの成功回数
fail：失敗回数
susp：保留回数

FACOM M200による
(a)拘束条件集合併合時ソート，リダクション時最小コスト拘束条件選定
(b)拘束条件集合併合時ソートのみ
(c)制御なし

【参考文献】

11)McDermott,D.:Circuit Design as problem solving, in Latombe,J.C.,(eds) Artificial Intelligence and Pattern Recognition in Computer Aided Design, North Holland,Amsterdam,pp.227-245 (1978).

12)Stallman,R.M. and Sussman,G.J. : Forward Reasoning and Dependency-Directed Backtracking in a System for Computer-Aided Circuit Analysis, Artificial Intelligence,Vol.9,pp.135-196 (1977).

13)Sussman,G.J. and Steele Jr,G.L.: CONSTRAINTS-A Language for Expressing Almost-Hierarchical Descriptions,Artificial Intelligence, Vol.14,pp.1-39 (1980).

14)Borning,A.:THINGLAB--An Object-Oriented System for Bulding Simulation using Constraints, Proc. 5th IJCAI, pp.497-498 (1977).

15)Gallaire,H.,Minker,J. and Nicolas J.M. : An overview and Introduction to Logic and Data Bases, in Gallaire,H.and Minker,J.(eds),Logic and Data Bases, Plenum,NewYork,pp.3-30 (1978).

16)Bundy,A. and Welham,B. : Using Meta-level Inferrence for Selective Application of Multiple Rewrite Rule Sets in Algebraic Manipulation,Artificial Intelligence, Vol.16,pp.189-212 (1981).

17)Shapiro,E.Y.: A Subset of Concurrent Prolog and Its Interpreter, ICOT Technical Report TR-003 (1983).

18)Winston,P.H.: Artificial Intelligence, Adisson-Wesley,Reading(Mass.) (1977).

10）長澤，古川，荒牧：ＡＤＬによる機械設計プログラム知識工学と人工知能研究会予稿集，３１－９（１９８３）．