ICOT Technical Report: TR-0781

TR-0781

ブール代数を用いた制約充足問題の定式化と解法についての検討

永井　保夫（東芝）、長谷川　隆三

June, 1992

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03)3456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

ブール代数を用いた制約充足問題の定式化と解法についての検討

Boolean Algebraic Approach to Constraint Satisfaction Problems

永井保夫
Yasuo NAGAI
(株)東芝 システム・ソフトウエア技術研究所
Toshiba Corp.

長谷川隆三
Ryuzou HASEGAWA
(財)新世代コンピュータ技術開発機構
ICOT

1 はじめに

制約充足問題は人工知能や画像理解の分野をはじめ，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のグラフの問題やパズルなどの探索問題，いわゆる組み合わせ問題を対象として研究が行われている[7),13)]．その代表的な解法としては，探索法や整合化手法を用いる方法があり[13),9),7)]，両者ともに探索の効率化を目的としておこなわれている[7),13)]．

一方，制約論理型言語は，制約という概念を論理型言語に導入することにより，すべての制約を宣言的に記述でき，宣言的なプログラミングを容易にする．制約論理型言語で取り扱う制約の対象領域には，有理数領域，実数領域，ブール代数領域，複素数領域，有限領域などがある[3)]．制約論理型言語において制約充足問題を取り扱った研究では，その特徴である探索機構を利用したアプローチ[6),8)]がとられている．これに対して，本研究では制約論理型言語のブール代数領域上の制約(これをブール制約という)の評価系を用いた代数的アプローチがとられる．

本稿では，ブール代数[14)]を用いて制約充足問題を定式化し，得られたブール方程式をブール制約とみなすことにより，ブール方程式の求解を制約論理型言語のブール制約評価系を用いておこなう方法について述べる．まず，第2章では，制約充足問題について述べ，ブール代数による定式化[12)]を説明する．第3章では，ブール代数による制約充足の解法を具体的な例題を用いて説明する．第4章では，制約論理型言語CAL[15)]のブール制約評価系において実現されているブーリアン・グレブナ基底による解法[16)]について説明する．第5章では，ブール制約評価系を用いた制約充足処理の効率化について検討する．特に，ブール制約(集合)のもつ構造情報を解析し，求められた解析情報を用いて制約評価系を効率化する方法について検討する．第6章では，制約論理型言語CALのブール制約を用いて制約充足問題を記述したプログラムを示し，いくつかの例題に対して同様の定式化をおこなう．第7章では，効率化手法の適用実験ならびに評価をおこない，さらに，効率化について考察する．

2 制約充足問題とブール代数による定式化

2.1 制約充足問題

制約充足問題とは，次のような変数の有限集合および各変数に対応する離散値の有限集合のドメインから値を選択し，すべての制約を満足するように各変数に対して値を割り当てる問題である．制約とは適用対象の構成要素およびその属性間で成立する関係を宣言的に記述したものである．[13),7)]

- 変数: $V = \{v_1, \ldots, v_n\}$
- ドメイン: $D_i = \{d_1, \ldots, d_m\}$, $i = 1, \ldots, n$, 各変数 v_n は離散値の有限集合 D_i から値 d_m を割り当てられる．
- 制約: 集合 $C = \{c_1, \ldots, c_l\}$,
  where $c_n \equiv (v_1, v_2, \ldots, v_n) \subseteq D_1 \times D_2 \times \ldots \times D_n$

n 個の変数に対する制約(n 項制約)は2項制約により表現できるので[5)]，今後は制約充足問題を考える場合には，2項制約のみを対象として検討する．

2.2 ブール代数による制約充足問題の定式化検討

2.2.1 ブール代数

ブール代数とは，0と1のみからなるブール集合 B 上で交換律，結合律，べき等律，吸収律，分配律，復元律，ド・モルガン定理，単位元律，普遍限界，補元律の10個の性質を満足する代数系 $< B, \wedge, \vee, \neg, 0, 1 >$ である [14]．

ブール多項式は，0ならびに1を含まない変数 $X_n = \{x_1, \cdots x_n\}$ に対して: a) $x_1, \ldots, x_n$，0，1はブール多項式である．b) p と q がブール多項式ならば，$p \vee q, p \wedge q, \neg p$ も同様にブール多項式である．を用いて再帰的に定義される．また，ブール多項式を0とおいたもの (=0) をブール代数方程式 (ブール方程式) という[1]．

2.2.2 ブール代数を用いた制約充足問題の定式化

次のように変数に関する制約と2つの変数間において成立する制約にわけて制約充足問題を定式化する．

(a) 変数に関する制約

変数集合を $V = \{v_1, \ldots, v_n\}$，各変数を v_i がとりうる離散値の有限集合 $D_i = \{d_1, \ldots, d_m\}, (i = 1, \ldots, n)$ とする．変数 v_i に対するドメイン D_i の要素 d_j の割り当てをブール方程式 $x_{ij} = 1$ により，割り当て禁止を $x_{ij} = 0$ によりあらわす．なお，以下，とくにことわらない限り，$i = 1, \ldots, n$ とする．

$$\bigvee_{j=1}^{m} x_{ij} = 1 \quad (1)$$

同時に，各変数 v_i はドメイン D_i からその要素である値 d_j をひとつだけとり，これをブール方程式として表現すると次のようになる．

$$\bigwedge_{1 \le j,k \le m, j \ne k} \neg x_{ij} \vee \neg x_{ik} = 1 \quad (2.1)$$

上式の両辺の否定をとり簡単化すると，

$$\bigwedge_{1 \le j,k \le m, j \ne k} x_{ij} \wedge x_{ik} = 0 \quad (2.2)$$

(b) 2つの変数間において成り立つ制約 (2項制約)

制約 C_{ij} は，変数集合 V の部分集合である変数 v_i と v_j を要素とする変数集合 V_{ij} とこれに対応した2項タプル集合 T_{ij} により表現される．

$$C_{ij} = (V_{ij}, T_{ij}), \quad where \ \ V_{ij} \subseteq V \quad (3)$$

この2項タプル集合は変数 v_i にドメイン D_i の要素を割り当て，変数 v_j にドメイン D_j の要素を割り当てた場合に可能な組み合わせならびに2変数間に対して割り当てを許さない値の組み合わせが考えられる．

前者の場合は，変数 v_i と変数 v_j のそれぞれのドメイン D_i と D_j の要素である d_k と d_l の組み合わせを2項タプル集合 $T_{ij} = \{(d_k, d_l) \mid d_k \in D_i \wedge d_l \in D_j\}$ とし，以下のブール方程式により表現する．

$$\bigvee_{(d_k,d_l) \in T_{ij}} x_{ik} \wedge x_{jl} = 1, \quad j = 1, \ldots, m \quad (4)$$

後者の場合は，変数 v_i と変数 v_j のそれぞれのドメイン D_i と D_j に対して値の割り当てを許さない d_k と d_l からなる組み合わせを2項のタプル集合 $\overline{T}_{ij} = \{(d_k, d_l) \mid D_i \times D_j -^2 T_{ij}\}$ とし，以下のブール方程式により表現する．

$$\bigwedge_{(d_k,d_l) \in \overline{T}_{ij}} \neg x_{ik} \vee \neg x_{jl} = 1, \quad j = 1, \ldots, m \quad (5.1)$$

上式の両辺の否定をとり簡単化すると，

$$\bigwedge_{(d_k,d_l) \in \overline{T}_{ij}} x_{ik} \wedge x_{jl} = 0, \quad j = 1, \ldots, m \quad (5.2)$$

3 具体例に対するブール代数を用いた制約充足問題の定式化

3.1 制約充足問題の具体例による説明

図1のような3つの変数 X, Y, Z とそのドメイン $D_X = \{5,2\}, D_Y = \{2,4\}, D_Z = \{5,2\}$ からなる制約ネットワークを例とした制約充足問題について説明する．制約は変数 X と Z 間の関係を示すタプル集合 $(X, Z) = \{(5,5), (2,2)\}$ と変数 Y と Z 間の関係を示すタプル集合 $(Y, Z) = \{(2,2), (4,2)\}$ により表現される．

変数 $X(= v_1)$ はそのドメインが $D_X = \{5,2\}$ であり，ドメインの要素からは値 d_j をひとつだけとるとすると，$X = v_1, D_X = D_1, d_1 = 5, d_2 = 2$ であり，ブール方程式により表現すると

[1] ブール多項式を1とおいたもの (=1) も同様にブール方程式とみなされる．

[2] このオペレータは集合差演算をあらわす．

$x_{11} \vee x_{12} = 1$, $\neg x_{11} \vee \neg x_{12} = 1$ が得られる.

同様に，変数 $Y(= v_2)$ と $Z(= v_3)$ に対しても，ブール方程式

$$x_{21} \vee x_{22} = 1, \quad \neg x_{21} \vee \neg x_{22} = 1$$
$$x_{31} \vee x_{32} = 1, \quad \neg x_{31} \vee \neg x_{32} = 1$$

が得られる.

さらに，2項制約 $(X, Z) = \{(5,5),(2,2)\}$ ならびに $(Y, Z) = \{(2,2),(4,2)\}$ に対する定式化では，次のように割り当てが許される値の組み合わせと割り当てが許されない値の組み合わせが考えられる.

a) 割り当てが許される値の組み合わせを考える場合

前者の場合の制約には $(X, Z) = T_{13}$, $(Y, Z) = T_{23}$ という関係があり，タプル集合には $T_{13} = \{(d_1, d_1),(d_2, d_2)\}$, $T_{23} = \{(d_1, d_2),(d_2, d_2)\}$ という関係が成り立つとする．この関係を用いて制約 (X, Z) と (Y, Z) のタプル T_{13} と T_{23} をブール方程式に変換すると

$$(x_{11} \wedge x_{31}) \vee (x_{12} \wedge x_{32}) = 1$$
$$(x_{21} \wedge x_{32}) \vee (x_{22} \wedge x_{32}) = 1$$

が求められる.

前者は変数 X と Z 間の制約をあらわし，後者は変数 Y と Z 間の制約を示す.

b) 割り当てが許されない (禁止される) 値の組み合わせを考える場合

変数 X と Z 間および変数 Y と Z 間において値の割り当てが許されないタプルをそれぞれ $\overline{T}_{13}$, $\overline{T}_{23}$ とすると $\overline{T}_{13} = \{(d_1, d_2),(d_2, d_1)\}, \overline{T}_{23} = \{(d_1, d_1),(d_2, d_1)\}$ となる．これをブール方程式により表現すると

$$(\neg x_{11} \vee \neg x_{32}) \wedge (\neg x_{12} \vee \neg x_{31}) = 1$$
$$(\neg x_{21} \vee \neg x_{31}) \wedge (\neg x_{22} \vee \neg x_{31}) = 1$$

または，

$$x_{11} \wedge x_{32} = 0, x_{12} \wedge x_{31} = 0$$
$$x_{21} \wedge x_{31} = 0, x_{22} \wedge x_{31} = 0$$

が得られる.

3.2 生成されるブール方程式と求められるべき充足解

割り当てが許される値の組み合わせと割り当てが許されない値の組み合わせという2つの場合が考えられ，たとえば，後者に対しては次のようなブール方程式集合 $BE1$ が生成される.

$$\left.\begin{array}{l} x_{11} \vee x_{12} = 1, \neg x_{11} \vee \neg x_{12} = 1 \\ x_{21} \vee x_{22} = 1, \neg x_{21} \vee \neg x_{22} = 1 \\ x_{31} \vee x_{32} = 1, \neg x_{31} \vee \neg x_{32} = 1 \\ (\neg x_{11} \vee \neg x_{32}) \wedge (\neg x_{12} \vee \neg x_{31}) = 1 \\ (\neg x_{21} \vee \neg x_{31}) \wedge (\neg x_{22} \vee \neg x_{31}) = 1 \end{array}\right\} = BE1$$

一般に，制約充足問題をブール代数を用いた定式化によりブール方程式に変換すると，方程式の数が非常に増えるため，規模の大きな方程式を解くことが必要となる．例えば，$BE1$ の場合には，$m \times n$ 個の変数 x_{ij} からなる $2n + |\overline{T}|$ 個の方程式が生成される.

最終的には，上記のブール方程式集合 $(BE1)$ を連立方程式として解くと，次のような充足解 $(x_{11}, x_{12}, x_{21}, x_{22}, x_{31}, x_{32}) = (0,1,0,1,0,1)$ と $(x_{11}, x_{12}, x_{21}, x_{22}, x_{31}, x_{32}) = (0,1,1,0,0,1)$ が得られる．前者は充足解が変数 X はドメイン D_X の2番目の要素2，Y は変数 Y のドメイン D_Y の2番目の要素4，Z はドメイン D_Z の2番目の要素2であることを示しており，同様に後者は $X = 2, Y = 2, Z = 2$ であることを示している．以下ではこのようなブール方程式をブール制約とよぶ.

4 ブール方程式解法による制約充足処理の効率化検討

ブール方程式 (制約) の解法は，数値処理や記号 (数式) 処理の分野において用いられている代数方程式解法に基づく．前者の数値処理分野においては，ガウス消去法に代表される直接法や繰り返し法に基づいた解法が提案されている [18]．一方，後者の記号処理分野においてはコンピュータ環論に基づき数式処理を利用した解法が提案されている [2], [15], [16].

4.1 代数制約解法の効率化手法の適用検討

われわれは，制約集合を制約グラフとして表現し，グラフ論的手法を用いて代数制約解法を効率化する手法について提案した [11]．本手法は，制約集合がもつ代数的構造を抽出し，この情報に基づいて求められた制約間の依存関係情報から，制約を効率よく解くための制約評価系に対する制御情報を生成し，効率的な解法を実現するものである.

しかしながら，制約の数と変数の数を比較して，どちらか一方が他方をおおきく上回る場合には，効率化手法はあまり有効ではない．ブール代数による定式化

では，ブール制約は効率化という観点から簡単な形式に変換されるが，一般にはブール制約が増加する傾向となる．つまり，取り扱う変数の数と比較して，制約の数が非常に多くなる．

以下では，このような問題点に対処するために，ブール方程式解法による制約充足処理の効率化手法として，制約充足問題のもつ構造情報を用いる方法ならびにブール制約の簡単化による方法について述べる．

4.2 制約ネットワークの構造情報を用いた効率化

制約充足問題の解法では，1) 木探索法，2) 整合化手法，3) 併合法がよく知られている [13)，7)]．制約ネットワークの構造情報を用いた効率化手法としては，木探索処理におけるバックトラックの改善手法や併合法が検討されている．

われわれは制約ネットワークにおいて決定される頂点の併合系列に基づくアプローチとして，併合法 [17)] と同様に制約ネットワークの頂点同志の整合性を保ちながら，単一の頂点をまとめる操作を繰り返し，求められた頂点併合系列に関する情報をもちいてブール制約の評価順序を決定し，制約評価を効率化する手法を提案する．

本効率化手法では，図2に示されるようにまず，制約ネットワークのグラフ表現に対して最も低い次数の頂点が選択され，次に制約の充足可能比 R [9)] をできるだけ小さくし (制約をきつくし)，次に選択すべき頂点がその時点で考慮すべき制約の数を減少させるように併合頂点が決定される．なお，最も次数の低い頂点の選択において複数の頂点が求められた場合には，それらの頂点の中で制約充足比 R の最も低い頂点が選択される．このように選択された頂点が順番に誘導され，頂点の併合順序が決定され，最終的には，この順序にしたがって縮退する頂点ならび枝に対応した制約集合の求解がおこなわれる．

4.3 ブール制約の簡単化による効率化

ブール多項式の乗法標準形はブール変数とその否定であるリテラル B_i を論理和演算 $\vee$ で結合したいくつかのブール多項式 $A_i(=\vee_{j\in J(i)}B_j)$ (ただし，$J(i)\subseteq\{1,\ldots,n\}$) を論理積演算 $\wedge$ で結合したブール多項式 $A_1\wedge\ldots\wedge A_n$ としてあらわされ，式の簡単化に用いられる．

ブール制約解法の効率化のために，ブール制約に対してブール代数の性質である分配律やド・モルガン定理を適用し，乗法標準形に変換し，ブール制約を簡単化する．この処理では，図3のアルゴリズムに示されるように，否定ならびに論理和演算を含んだ制約をできるだけ簡単な制約に変換する．このような簡単化が容易におこなわれるように，2項制約は禁止条件である割り当ての許されない場合として定式化される．図1の制約ネットワーク問題では，2項制約が禁止条件として表現された定式化からブール制約集合 $BE1$ が得られ，簡単化がおこなわれる．たとえば，この場合にはブール制約 $(\neg x_{11}\vee\neg x_{32})\wedge(\neg x_{12}\vee\neg x_{31})=1$ は乗法標準形であり，$A_1\wedge\cdots\wedge A_n=1$ は $A_1=1,\ldots,A_n=1$ と等価なので，$\neg x_{11}\vee\neg x_{32}=1,\neg x_{12}\vee\neg x_{31}=1$ が得られる．最終的にはド・モルガンの定理を適用して求められたブール制約集合 $x_{11}\wedge x_{32}=0, x_{12}\wedge x_{31}=0$ の求解がおこなわれる．

5 制約論理型言語のブール制約評価系

制約論理型言語は制約に関する表現方法ならびに制約をどのように解くかという問題解決技法を非決定的処理と単一化機構をもつ論理型言語に組み込んだ宣言的な言語である [3)，15)]．その構造は全体の処理を制御する推論機構と制約を解いて答えを求める制約評価系からなる．

5.1 ブーリアン・グレブナ基底を用いたブール制約評価系

制約論理型言語 CAL では，複素数領域を対象としたグレブナ基底を求め代数制約評価系ならびにブール代数領域を対象としたグレブナ基底 (ブーリアン・グレブナ基底) を求めるブール制約評価系が提供されている [15)，16)]．

ブール制約評価系では，ブール領域に対して拡張されたブッフバーガ・アルゴリズム [2)] に基づき，項書き換えがおこなわれ，ブーリアン・グレブナ基底が求められる．この処理は項書き換え操作とみなせるので，停止性ならびに合流性を同時にもち (すなわち完備であり)，概要は次のようになる：まず，入力である等式集合 E のすべての要素に対して項の順序付けをおこない，得られた書き換え規則集合を R とする．この R に含まれるすべての書き換え規則に対して要対を調べ，完備な項書き換え集合を生成する．このよ

うにして得られる要対集合を E とする．そして，要対集合 E から要素を選び，項の順序付けをおこない新たな書き換え規則を作る処理を，E が空になるまで繰り返しおこなう．このアルゴリズムは，グレブナ基底を求めるブッフバーガ・アルゴリズムと比較して，S-多項式処理に加えて自己要対式の処理が付け加えられている点が大きく異なる．

ブーリアン・グレブナ基底によるブール方程式解法アプローチは，従来の解法 [14] とは異なり，1) プログラムやゴール中で陽に記述されていない補助変数は導入する必要がなく，2) すべての制約は効率的な計算がおこなえる正規形をもつという特徴を有する [16]．このような特徴からブール制約の正規形 (基底) を用いてブール制約集合の充足不能性が判定できる．つまり，ブーリアン・グレブナ基底が存在しない場合には解が存在しないことを示し，基底が存在する場合には解の存在を示す．したがって，求められた基底は解を表現するので，健全性と完全性が保証されることになる．

5.2 ブーリアン・グレブナ基底を用いたブール制約評価系の効率化検討

ブール制約の求解法のひとつである制約論理型言語 CAL のブール制約評価系を用いた制約充足の効率化について検討する．CAL のブール制約評価系では，代数制約評価系と同様に変数に対して優先度を指定する機能が組み込み述語として提供されている．変数に対する優先度の指定により，単項間の順序を決定できるので，これに基づきブール制約評価系に対する書き換え規則が設定できる．さらに，制約の並べ変えにより，制約の評価順序が制御できる [15]．

ところで，ブール代数 $< B, \wedge, \vee, \neg, 0, 1 >$ は単位元をもつブール環 $< R, +, \cdot, 0, 1 >$ と同等であることが知られている [14]．ブール制約評価系では，ブール制約がブール環上の多項式に変換され，これらのブール環上の多項式に対するグレブナ基底が求められる．このブール環では，ブール代数の性質である交換律，結合律，分配律ならびに単位元律が満足されるのに加えて，$x + x = 0$ ならびに $x \cdot x = x$ という 2 つの公理が成立する．また，ブール代数上の演算 $\wedge, \vee, \neg$ はブール環 $< R, +, \cdot, 0, 1 >$ によって，$x \vee y =_{def} x + y + x \cdot y$, $x \wedge y =_{def} x \cdot y$, $\neg x =_{def} 1 + x$ と表現される [14]．このようなブール演算 $\wedge, \vee, \neg$ によって表現されたブール制約が複雑であれば，変換されるブール環上の多項式も一層複雑な形式をとることが考えられる．そこで，ブーリアン・グレブナ基底を求める処理を効率化するためには，ブール環上の多項式が複雑な形式をとらないように，ブール制約を簡単な形式で与えられることが望ましい．前処理である簡単化処理によりブール制約を簡単な形式に表現できれば，制約評価の効率化が期待できる．そこで，簡単な形式に変換できる制約は，できるかぎり簡単化する．

また，制約の評価順序は制約が簡単な形式をとるものほど，その評価順序をはじめに設定し，複雑な形式の制約ほど評価順序をあとのほうに設定するほうが効率化が期待できる．たとえば，論理和演算からなる多項式 $x \vee y$ は $x + y + x \cdot y$ というブール環上の多項式に変換され，同様に $x \vee y \vee z$ は $x + y + x \cdot y + z + (x + y + x \cdot y) \cdot z$ という多項式に変換される．ブール制約評価系では，変換されたブール環上の多項式に対するグレブナ基底を求めることになるので，論理和演算 $\vee$ を含んだ制約はなるべくあとで評価することが望ましいと考えられる．

以上の点から，ブーリアン・グレブナ基底を求める制約評価系の効率化手法として，1) ブール制約の簡単化，2) 制約充足問題がもつネットワークの構造情報を用いた制約の評価順序の決定の 2 つの項目を取り上げる．なお，制約の評価順位は，制約ネットワークにおいて 2 項制約を禁止条件として定式化した場合に，頂点の次数およびタプル集合 $\overline{T}(= D_i \times D_j - T_{ij})$ から求められる制約充足比 R により決定される．

6 制約論理型言語 CAL のブール制約を用いた記述ならびに効率化手法の適用

本章では，制約論理型言語 CAL のブール制約を用いて記述したプログラムを示し，いくつかの例題に対して定式化をおこなう．さらに，上記に示された 2 種類の効率化手法を適用した例について示す．

6.1 制約ネットワーク問題

図 1 に示されている例題ならびに 2 種類の制約ネットワークからなる 3 種類の問題を制約論理型言語 CAL [15] で記述し，ブール制約評価系を利用して制約充足をおこなう．

c_network/6 は制約ネットワークにより表現された例題を CAL により記述したプログラムである．このプログラムはブール制約の簡単化の容易性を考慮して，変数に関する制約は禁止制約として定式化した．なお，

bool: X11 \/ X12 =1 はブール制約をあらわし，\/ は論理和 $\vee$ を，& は論理積 $\wedge$ を，~X11 はブール変数 X11 の否定をそれぞれあらわす．

```
:- public c_network/6.

c_network(X11,X12,X21,X22,X31,X32) :-

        bool: X11 \/ X12 =1,
        bool: X21 \/ X22 =1,
        bool: X31 \/ X32 =1,
        bool: ~X11 \/ ~X12 =1,
        bool: ~X21 \/ ~X22 =1,
        bool: ~X31 \/ ~X32 =1,
        bool: (~X11 \/ ~X32) & (~X12 \/ ~X31) =1,
        bool: (~X21 \/ ~X31) & (~X22 \/ ~X31) =1.
```

プログラム c_network/6 を実行した結果，次のようなブール変数 x_{21} のみからなるブーリアン・グレブナ基底が求められる．なお，$\overline{x}_{21}$ はブール変数 x_{21} の否定形をあらわす．

$$x_{11}=0, x_{12}=1, x_{22}=\overline{x}_{21}, x_{31}=0, x_{32}=1$$

最終的な解としては，ブール変数 x_{21} に 0 と 1 を割り当てた場合，

$$x_{11}=0, x_{12}=1, x_{21}=0, x_{22}=1, x_{31}=0, x_{32}=1$$

ならびに，

$$x_{11}=0, x_{12}=1, x_{21}=1, x_{22}=0, x_{31}=0, x_{32}=1$$

が求められる．

次に，ブール制約が簡単化されたプログラム c_networkn/6 を示す．簡単化処理では，まずプログラム c_network/6 中のブール制約，
たとえば bool : ~X11\/~X12 = 1 が bool : X11 & X12 = 0 に変換される．また，制約
bool : (~X11\/~X32) & (~X12\/~X31) = 1 は
bool : X11 & X32 = 0, bool : X12 & X31 = 0 に変換される．

```
:- public c_network/6.

c_networkn(X11,X12,X21,X22,X31,X32) :-

        bool: X11 \/ X12 =1,
        bool: X21 \/ X22 =1,
        bool: X31 \/ X32 =1,
        bool: X11 & X12 =0,
        bool: X21 & X22 =0,
        bool: X31 & X32 =0,
        bool: X11 & X32 =0,
        bool: X12 & X31 =0,
        bool: X21 & X31 =0,
        bool: X22 & X31 =0.
```

さらに，以下の c_networknn/6 は制約ネットワークの構造情報を用いて決定された頂点の併合順序 (ここでは $X \Rightarrow Z \Rightarrow Y$ とした) に基づき，ボディー部のサブゴールである制約を並び換えたプログラムである．頂点の併合順序は図 2 のアルゴリズムにより決定される: 図 1 の制約ネットワークの各頂点の次数は変数 X が 1，Y が 1，Z が 2 となるので，まず最小次数の頂点が併合頂点として選ばれる．該当する頂点には X と Y があるので，それぞれの頂点を変数に含む制約集合を求め，それらの要素である 2 項制約の充足可能比の和が最小となる頂点を選択する．上の例では，C_{XZ} と C_{YZ} の充足可能度 R_{XZ} と R_{YZ} がそれぞれ 1/2 となり，両方の頂点が選択候補となるが，ここでは X を併合頂点として選択する．さらに，制約ネットワークから頂点 X が取り除かれ縮退したネットワークから同様な操作により頂点 Y，さらに頂点 Z が選択され，最終的に $X \Rightarrow Z \Rightarrow Y$ という併合順序が求められる．

```
:- public c_networknn/6.

c_networknn(X11,X12,X21,X22,X31,X32) :-

        bool: X11 & X12 =0,
        bool: X11 \/ X12 =1,
        bool: X12 & X31 =0,
        bool: X11 & X32 =0,
        bool: X31 & X32 =0,
        bool: X31 \/ X32 =1,
        bool: X21 & X31 =0,
        bool: X22 & X31 =0,
        bool: X21 & X22 =0,
        bool: X21 \/ X22 =1.
```

6.2 クイーン問題

ここでは，制約充足問題の代表的な例題として N-クイーン問題を取り上げる．

N-クイーン問題では，変数 v_i が盤の $1 \leq n \leq n$ 列に対応し，それぞれの変数 v_i がドメインを $D_i = \{1,\ldots,n\}$ を示し，その要素が盤の行をあらわす．クイーンを i 列の j 行目に置くことは変数 v_i への値 j の

割り当てとみなされ，ブール制約 $x_{ij} = 1$ により表現される．

以下では，4クイーン問題を2項制約を用いた制約充足問題とみなして定式化をおこなう [9]．任意の2つのクイーンが互いに攻撃しあわないという関係は $(v_i \neq v_j) \wedge (|v_i - v_j| \neq j - i)$ (但し，$1 \leq i < j \leq n$) を満足する2項制約として表現される．一般に，N-クイーン問題は ${}_nC_2 = n(n-1)/2$ 個の2項制約を用いたネットワークとしてあらわされる．4クイーン問題は，図4のように6個の2項制約からなる制約ネットワークとして与えられる [9]．4クイーン問題に対応する制約ネットワークは，図5のように4個のノードからなる完全グラフ K_4 [1] により表現される．各ノードは変数に関する制約に対応する．たとえば，ノード N_1 は変数 v_1 に関する制約集合 $\{C_{10}, C_1\}$ に対応する．なお，C_{10} は変数に関する制約において(1)式として表わされた制約を示し，C_1 は(2.2)式として表わされた制約を示す．ノード N_1 と N_2 の間のエッジ C_{12} は2つの変数 v_1 と v_2 間において成立する2項制約 C_{12} に対応する．図5に示された2項制約により表現された4クイーン問題は，ブール代数を用いて定式化され，制約論理型言語のブール制約を用いたプログラムとして記述される．

次に，4クイーン問題に対する2種類の効率化手法の適用について説明する．まず，制約簡単化処理により，上記のプログラムのブール制約が簡単化される．さらに，ブール制約の簡単化されたプログラムに対して，上記の制約集合の構造情報を用いて決定された頂点の併合順序に基づき，制約の評価順序が決定される．ここでは，プログラム中のブール制約が論理和形式をとる場合には，ブール制約はブール環上のより複雑な多項式へ変換された後に，グレブナ基底が計算されるので，簡単化されるプログラムのボディ部では，簡単なブール環上の多項式に変換される制約をできるだけ先に評価し，複雑なブール環上の多項式に変換される制約をできるだけ後に評価するようにゴールの順序を決定する．頂点の併合順序は図2のアルゴリズムにより決定される．図6の制約ネットワークは完全グラフ K_4 として表現されるので，各頂点 v_1, v_2, v_3, v_4 の次数 (すべての次数が3) が求められる．そして，各々の頂点を変数に含む制約集合を求め，それらの要素である2項制約の充足可能比 R の和が最小となる頂点を選択する．頂点 v_1 を変数にする2項制約の集合は $\{C_{12}, C_{13}, C_{14}\}$ となる．併合の選択候補の頂点は v_1 と v_4 であり，ここでは頂点 v_1 を併合頂点とする．そして，制約ネットワークから頂点 v_1 を取り除くことにより縮退したネットワークに対し同様の操作を繰り返し，最終的に $v_1 \Rightarrow v_2 \Rightarrow v_3 \Rightarrow v_4$ という併合順序が決定され，図6の頂点併合グラフが得られる．

7 実験結果および考察

7.1 実験結果

2項制約により表現された制約充足問題をブール制約を用いて定式化し，制約論理型言語 CAL のブール制約評価系を用いて実験をおこなった．実験結果は図7から図10に示される．実験は逐次推論マシン PSI-II [10] 上で，制約論理型言語 CAL バージョン1.4を用いた．

今回は，表1に示された3種類の制約ネットワークにより表現された制約充足問題，4クイーンならびに5クイーン問題を対象として実験をおこなった．4クイーンならびに5クイーン問題では，全解ならびに単解 (部分解) を求めた．本実験では問題に対する解としてブーリアン・グレブナ基底を求めているが，制約集合の要素の数が多い場合にはその処理が非効率になる傾向がある．ここでは，ブーリアン・グレブナ基底をそのまま求める解法ならびに，いくつかのブール変数に対してあらかじめ0または1の値を割り当てできるだけ制約を簡単化しながら制約評価により基底を求める解法を適用した．前者を「記号による解法」とよび，後者を「数値割り当てによる解法」とよぶ．「数値割り当てによる解法」は4クイーンならびに5クイーン問題に対してのみ適用した．

ブール制約評価系の効率化手法として，次のような処理をおこなった．まず，ブール代数による定式化により得られたブール制約集合を CAL プログラムとして記述し，実行する場合を「効率化手法の適用前」とする．否定ならびに論理和演算を含まないようにブール制約が簡単化されたプログラムを実行する手法を「制約簡単化」とする．表2は各問題に対する「制約簡単化」の適用結果を示す．そして，制約ネットワークの構造情報を用いて制約の評価順序を決定し，その順序に基づいて簡単化された制約の評価順序を修正し，プログラムを実行する場合を「制約並び換え」とする．なお，それぞれの実験で求められた実行時間には，簡単化ならびに制約の評価順序決定に関する処理時間は含まれていない．

図7から図10にはそれぞれの解法に対して効率化

手法を適用した結果が示される．制約ネットワーク問題に対しては，効率化手法である「制約簡単化」ならびに「制約並び換え」を適用した「記号による解法」が有効であることがわかった．たとえば，前者の「制約簡単化」においては約30%から40%の処理時間の改善がみられた(図7)．4クイーンならびに5クイーン問題においては，「制約簡単化」ならびに「制約並び換え」を適用した「数値による解法」が効率化手法として非常に有効であることがわかった．たとえば，4クイーン問題全解問題の「数値による解法」では，「制約簡単化」により約6.4倍の処理時間の改善がみられ，「制約並び換え」により約10.5倍の処理時間の改善がみられた(図7)．また，制約ネットワーク問題や4クイーン問題と比較して規模の大きな5クイーン全解問題の「数値による解法」では，「制約簡単化」により約218倍の処理時間の改善がみられ，「制約並び換え」により約326倍もの処理時間の改善がみられた(図9)．しかしながら，4クイーン問題に対する「記号による解法」では，「制約簡単化」において約35%の処理時間の改善がみられたが，「制約並び換え」においては改善がほとんどみられなかった(図8)．また，5クイーン全解問題に対して「制約簡単化」を適用した「記号による解法」では，「数値による解法」の場合ほどではないが，約11.5倍の処理時間の改善がみられ有効であった(図8)．一方，「制約並べ換え」では「制約簡単化」の場合と比較して，処理時間において約46%の改善にとどまった(図9)．

7.2 考察

制約充足問題の代表的な解法である探索法とブーリアン・グレブナ基底を求める制約評価系を用いた方法には次のような違いがある．探索法における解は，列挙された形式をとる．これに対して，ブーリアン・グレブナ基底を求める制約評価系を用いた方法による解は，ブール変数間の関係をあらわす基底形式であり，列挙解とはなっていない．探索法による解を得るためには，6.1節において求められたグレブナ基底の変数に対してブール値(0または1)を割り当て，解を列挙することが必要である．「記号による解法」では，ブール制約集合のグレブナ基底を解として求める．一方，「数値による解法」では，あるブール変数に対するブール値1または0の割り当てにより，簡単化可能な制約はできる限り簡単化して制約評価をおこないグレブナ基底を求める．この「数値による解法」は，制約充足問題においていくつかの変数に対してドメインの要素を割り当てて解を求めることに相当する．

次に，効率化手法である「制約簡単化」ならびに「制約並び換え」において考慮すべき点について述べる．「制約簡単化」では，生成される制約数の増加による制約評価時間の増加と制約の簡単化に要する時間とのトレード・オフを考慮することが必要である．これは，ブール制約の簡単化では，制約の数の増加にしたがい，非常に多くの制約をあらたに生成するためである．たとえば，$\neg P_1 \vee \cdots \vee \neg P_n = 1$のような負リテラルからなる制約の簡単化では，変換により否定ならびに論理和演算が取り除かれ，論理積のみからなる制約$P_1 \wedge \cdots \wedge P_n = 0$があらたに生成される．また，$P_1 \wedge \cdots \wedge P_n = 1$のような制約を変換する場合には，$P_1 = 1, \ldots, P_n = 1$のように多くの制約が生成される．一方，「制約並べ換え」では，制約ネットワークの解析時間と制約評価にかかる時間とのトレード・オフについて考慮することが必要である．制約ネットワーク解析では，制約ネットワークの頂点の次数ならびに制約充足可能比に基づき，頂点の併合順序が決定されている．そこでは，探索処理の効率化手法において提案された制約の充足可能比という尺度を用いられ，ネットワークの頂点に対応するブール変数の数ならびに枝に対応する2項制約(すなわちタプル)の数を考慮した解析がおこなわれているが，その有効性についてはより詳細な検討が必要である．

さらに，効率化手法である「制約簡単化」ならびに「制約並び換え」の有効性について，5.1節のブーリアン・グレブナ基底計算アルゴリズムと関連づけて考察する．「制約簡単化」では，入力として与えられた制約の簡単化により，求められた制約の書き換え規則を用いた項書き換え操作により制約評価がおこなわれる．実験結果から示されるように，制約ネットワーク問題ならびにクイーン問題のいずれの問題に対しても「制約簡単化」は有効な手法であった．これは，前処理である「制約簡単化」により，項書き換え処理から求められる書き換え規則の複雑化を抑制し，グレブナ基底計算に要する時間を減らす効果があったからであると考えられる．また，「制約並び換え」では，制約ネットワークの構造解析により変数間の共有関係をもつ制約が集められ項書き換え処理により制約評価がおこなわれる．4クイーンならび5クイーン問題に対して「制約簡単化」ならびに「制約並び換え」を適用した「数値による解法」では，処理時間のおおきな改善がみられた．これは「数値による解法」では変数へ

の0または1の割り当てにより簡単化できる制約はできるだけ簡単化され，変数間の共有情報を有効に利用して項書き換え操作が実行されたからであると考えられる．一方，4クイーンならび5クイーン問題に対して「制約簡単化」ならびに「制約並び換え」を適用した「記号による解法」では，処理時間の改善がほとんどみられなかった．これは「記号による解法」において「制約並び換え」をおこなっても，項書き換え操作により求められる書き換え規則の複雑化を抑えることができなかったことが原因で基底計算に時間を要したからであると考えられる．

最後に，提案した効率化手法とフォワードチェック法との関連について述べる．フォワードチェック法[6]は，制約ネットワークの整合化に基づき従来のバックトラック探索を高度化するように拡張された探索手法の一種であり，制約論理型言語CHIPに取り入れられている[6]．ブール制約評価系を用いた制約充足処理の効率化手法のなかで，「制約並び換え」を適用した「数値による解法」は，フォワードチェック法を実現していると考えられる．本手法では，制約が静的に与えられている場合と同様に，制約の動的生成や追加により制約充足がおこなわれる場合には対処可能である．これはブール制約評価系のアルゴリズムがインクリメンタルに制約を評価できるためである．しかしながら，制約の除去という動的な取り扱いにはそのままでは対処できない．

本実験では制約ネットワーク問題ならびにクイーン問題という限られた問題を対象としたが，今後は，より規模の大きな問題(制約の数が多くかつ制約が複雑な形式をとる問題と制約ネットワークの構造が複雑な問題)に対する適用実験とその有効性の詳細な検討が必要であると考える．

8 おわりに

ブール代数領域を対象とした制約を取り扱う制約論理型言語を用いて，制約充足問題に対する従来の探索を基本とした解法とは異なるあらたな代数的な解法を提案した．具体的にはいくつかの例題を取り上げて定式化をおこない，ブール制約を解くために，制約論理プログラミング言語CALのブール制約評価系を用いた．その結果，制約充足問題の解法として探索法がよく知られているが，それとは別にブール代数評価系を用いた解析的な方法が有効であることを示した．さらに，制約評価系の効率化をはかるために，制約集合のもつ構造情報を用いた手法について検討し，その有効性を示した．

謝辞

本研究は第5世代コンピュータプロジェクトの一環として行なわれた．本研究の機会を与えてくださり，常にご指導いただいたＩＣＯＴの淵一博所長，古川康一研究次長，生駒憲治部長代理(現NTTデータ通信(株))，新田克己第7研究室室長ならびに，有益なコメントをいただいた相場亮第4研究室室長代理に深く感謝いたします．また，CALを利用するにあたり，いろいろ教えて頂いた第4研究室の皆様に感謝いたします．本研究の機会を与えていただいた(株)東芝システム・ソフトウエア技術研究所の西島誠一所長ならびに河野毅部長に深謝いたします．

参考文献

1) Beineke, L. W. and Wilson, R. J. (Eds.) : *Selected Topics in Graph Theory 2*, Academic Press (1983).
2) Buchberger, B. : Gröbner bases: An Algorithmic Method in Polynomial Ideal Theory, In *Multidimensional Systems Theory* (ed. Bose, N.), pp.184-232, D. Reidel Publ. Comp., Dordrecht (1985).
3) Cohen, J. : Constraint Logic Programming Languages, *Comms. ACM*, Vol.33, No.7, pp.52-68 (1990).
4) de Kleer, J. : A Comparison of ATMS and CSP Techniques, In *Proc. of the IJCAI-89*, pp.290-296 (1989).
5) Freuder, E. C. : Synthesizing Constraint Expressions, *Comms. ACM*, Vol.21, No.11, pp.958-966 (1978).
6) Hentenryck, P. V. : *Constraint Satisfaction in Logic Programming*, MIT Press (1989).
7) Mackworth, A. K. : Constraint Satisfaction, In *Encyclopedia of Artificial Intelligence ((ed.) Shapiro, S.C.), Vol.1*, pp.205-211, John Wiley & Sons, Inc. (1987).
8) Montanari, U. and Rossi, F. : Perfect Relaxation in Constraint Logic Programming, In *Proc. of ICLP '91*, pp.223-237 (1991).
9) Nadel, B. A. : Representation Selection for Constraint Satisfaction: A Case Study Using n-Queens, In *Proc of IEEE Expert*, pp.16-23, June (1990).

10) Nakashima, H. and Nakajima, K.: Hardware architecture of the sequential inference machine: PSI-II, In *Procs. of 1987 Symposium on Logic Programming*, pp.104-113 (1987).
11) 永井保夫： 制約グラフの構造分解および整合性解析による代数制約評価系の効率化についての検討，人工知能学会研究会資料 SIG-FAI-HICG-KBS-9001-12 (1990).
12) 永井保夫： ブール代数を用いた制約充足問題の定式化とその解法についての検討，電子情報通信学会研究会資料 AI90-73 (1991).
13) 西原清一：制約充足問題 (CSP) の基礎と動向，1991 年度人工知能学会全国大会 (第 5 回) チュートリアル資料 B-1 (1991).
14) Rudeanu, S.: *Boolean Functions and Equations*, North-Holland Publ. Comp. (1974).
15) Sakai, K. and Aiba, A.: CAL: Theoretical Background of Constraint Logic Programming and its Application, In *J. of Symbolic Computation*, Vol. 8, pp.589-603 (1989).
16) Sakai, K. and Sato, Y.: Application of Ideal theory to Boolean constraint solving, In *Proc. of PRICAI '90*, pp.490-495 (1990).
17) 塩澤恒道，西原清一，池田克夫: 拘束条件の構造を考慮した整合ラベリング問題の解法，情報処理学会論文誌， Vol. 27, No. 10, pp.927-935 (1986).
18) 津田孝夫：岩波講座 ソフトウエア科学 9, 数値処理プログラミング，岩波書店 (1988).

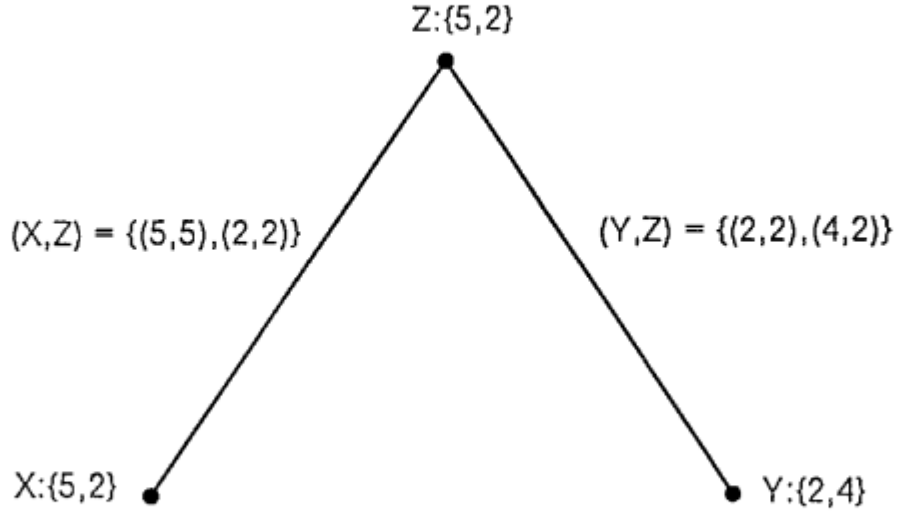

図 1　制約ネットワーク 1
Fig. 1 Example of constraint network no.1

```
procedure vertex_merging(G(V,E): input, Vertices: output)
1     G_0 — G(V,E);
2     i — 0;
3     Vertices — 0;
4     while G_i ≠ 0 do
5        for every v ∈ V
6          Pick a subset DV of V with a minimum degree;
7          for every dv ∈ DV
8            Adj_dv — {(dv,w) ∈ E such that w ∈ V}
9            R_dv — Σ_{i∈Adj_dv} R_dv,i;
10         endfor
11       Pick a vertex v_i with a minimum R;
12       endfor
13       Vertices — Vertices ∪ {v_i};
14       G_{i+1} — G(V \ {v_i}, E(V \ {v_i}));
15       i — i + 1;
16    endwhile
17    end.
```

図2　頂点併合アルゴリズム
Fig. 2　Vertex merging algorithm

```
procedure transform(Constraint: input, Constraint' : output)
1     Constraint'.Constraint'' — {} :
2     case Constraint of
3        A1 ∧ ··· ∧ An = 1 :
4           for i=1 to n do
5             begin
6               transform(Ai = 1, Constraint'');
7               Constraint' — Constraint' ∪ Constraint'';
8             end;
9        ¬A1 ∨ ··· ∨ ¬An = 1:
10          Constraint' — Constraint' ∪ {A1 ∧ ··· ∧ An = 0};
11       ¬A1 ∨ ··· ∨ ¬An = 0:
12          Constraint' — Constraint' ∪ {A1 ∧ ··· ∧ An = 1};
13       ¬A = 1:
14          Constraint' — Constraint' ∪ {A = 0};
15       ¬A = 0:
16          Constraint' — Constraint' ∪ {A = 1};
17    end.
```

図3　ブール制約の簡単化アルゴリズム

Fig. 3　Algorithm of boolean constraint transforamtion

表1　実験において使用される問題の特徴

Table 1　Characteristics of experimental problems

定式化前の問題	ノード数	アーク数	タプル数
ネットワーク1	3	2	4
ネットワーク2	3	3	7
ネットワーク3	4	4	8
4クイーン問題	4	6	44
5クイーン問題	5	10	140

表2　ブール代数による定式化後ならびに簡単化処理後の問題の特徴

Table 2　Result of boolean formalization of problems and transformation of boolean constraint

定式化後の問題	ブール変数の数	ブール制約の数	簡単化後の制約の数
ネットワーク1	6	8	10
ネットワーク2	6	8	14
ネットワーク3	8	8	12
4クイーン問題	16	34	80
5クイーン問題	25	65	165

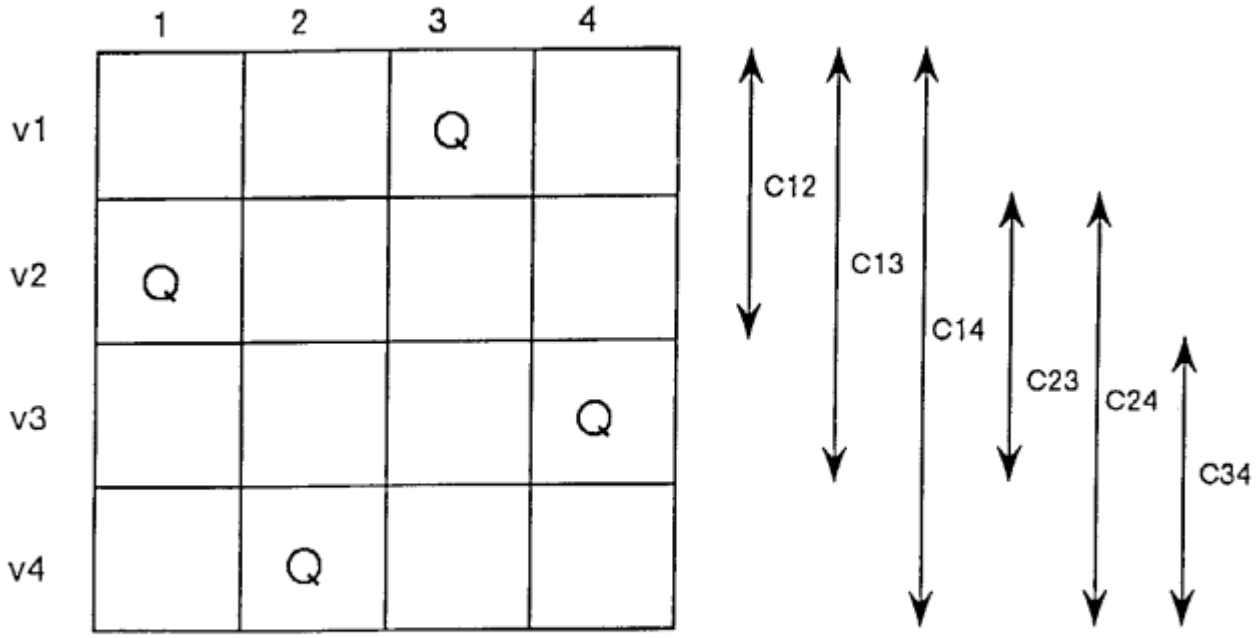

図４　2項制約による４クイーン問題の定式化
Fig. 4 Formalization of 4-queen problem using binary constraint representation

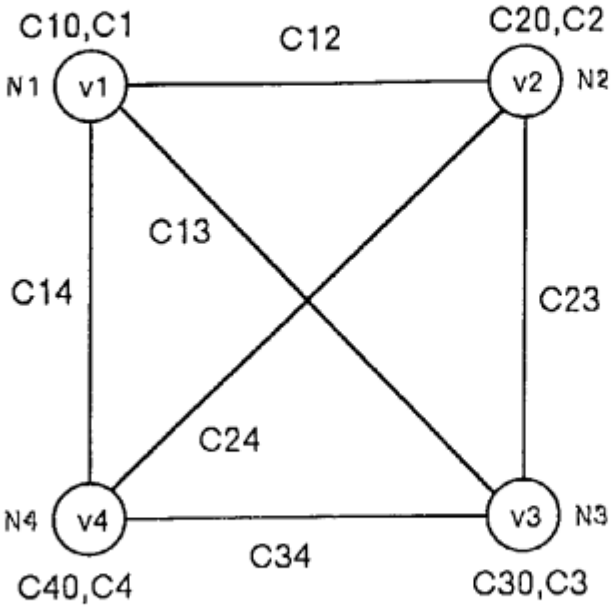

図５　制約ネットワーク（４クイーン問題）
Fig. 5 Constraint network representation corresponding to 4-queen problem

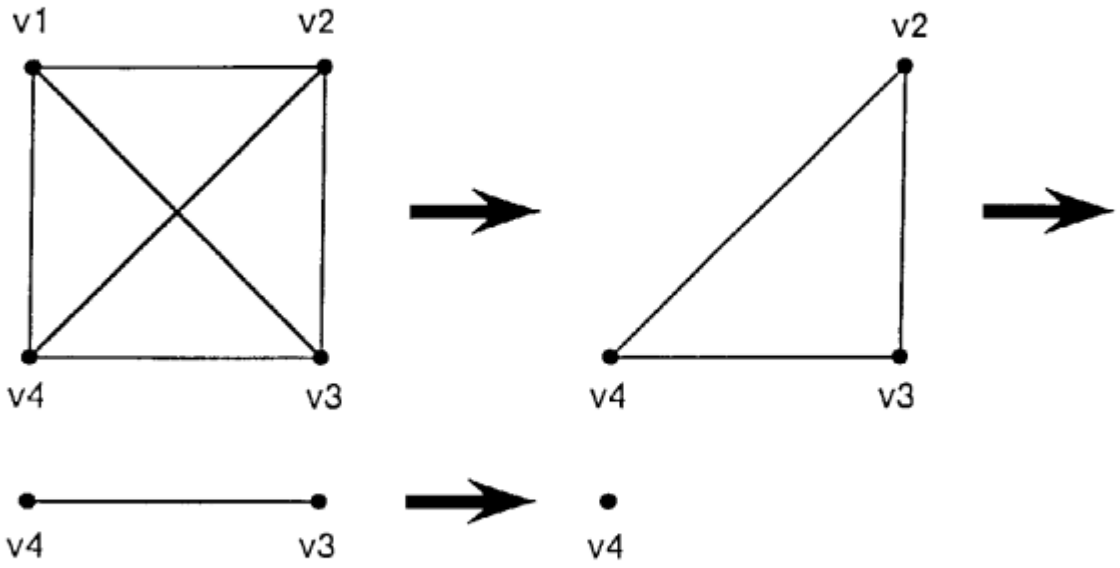

図 6　制約ネットワークの構造情報より求められる変数の選択順序
Fig. 6 Sequence of vertex selection determined from constraint network

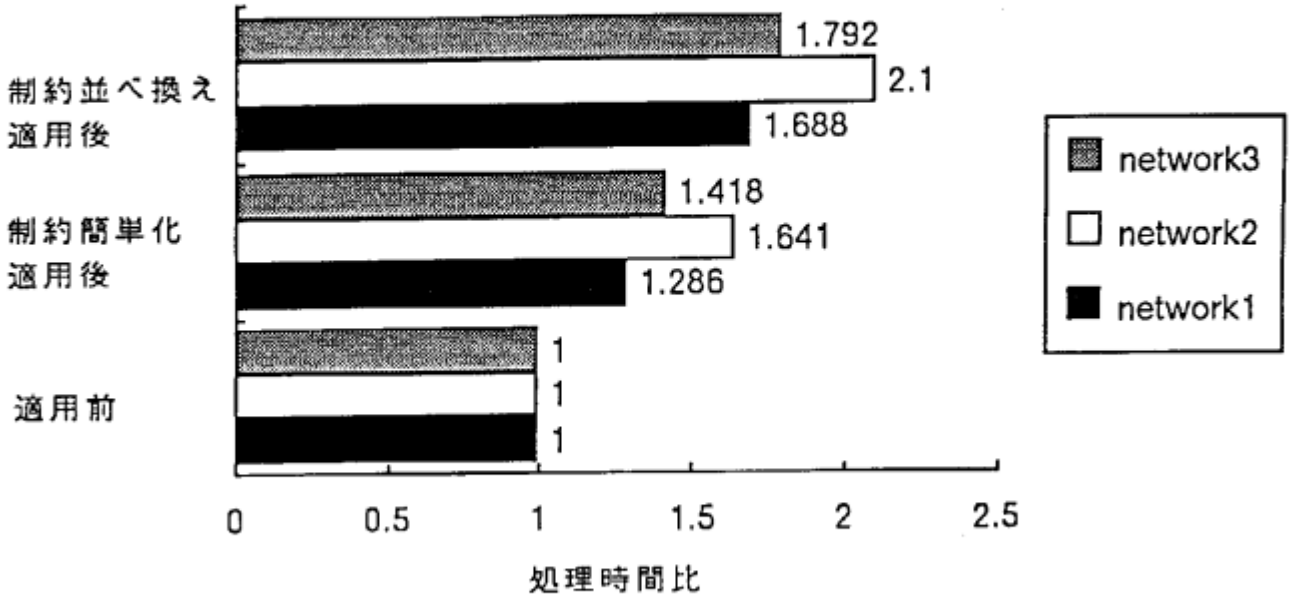

図 7　効率化手法の適用による処理時間の改善（制約ネットワーク問題）
Fig. 7 Performance improvement of constraint network problems

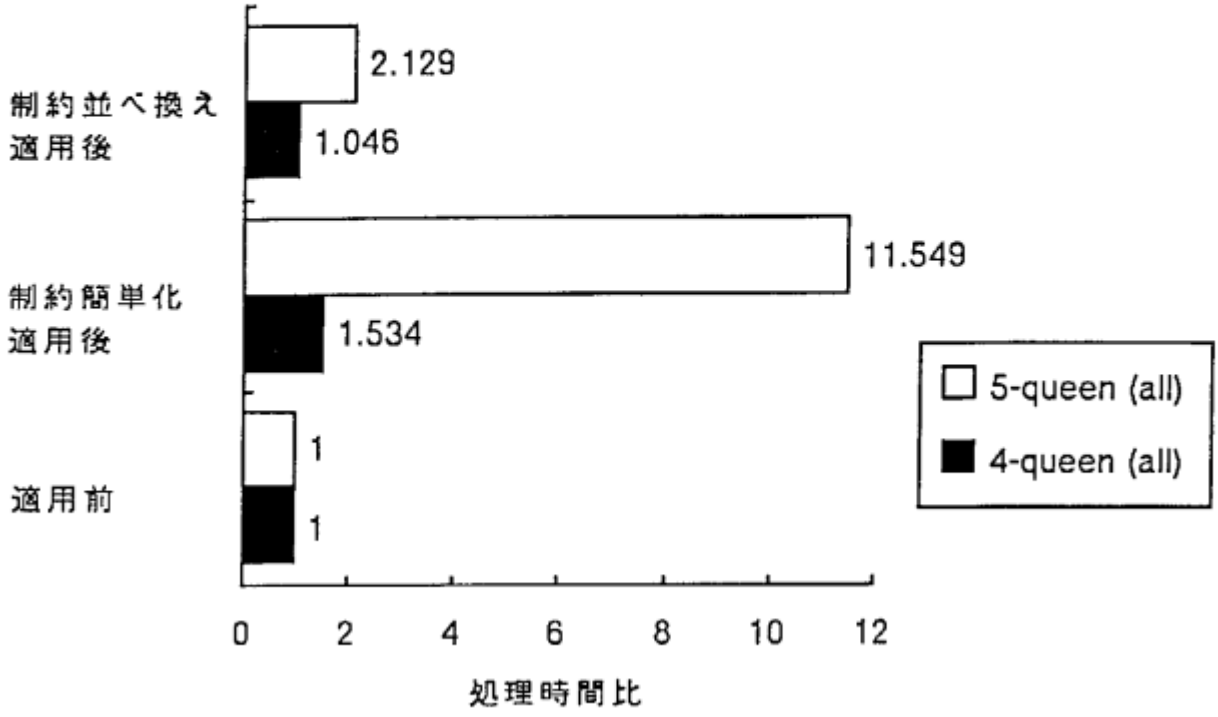

図 8　効率化手法適用による記号解法の処理時間の改善（クイーン問題の全解）
Fig. 8 Performance improvement of n-queen problems (all solutions)

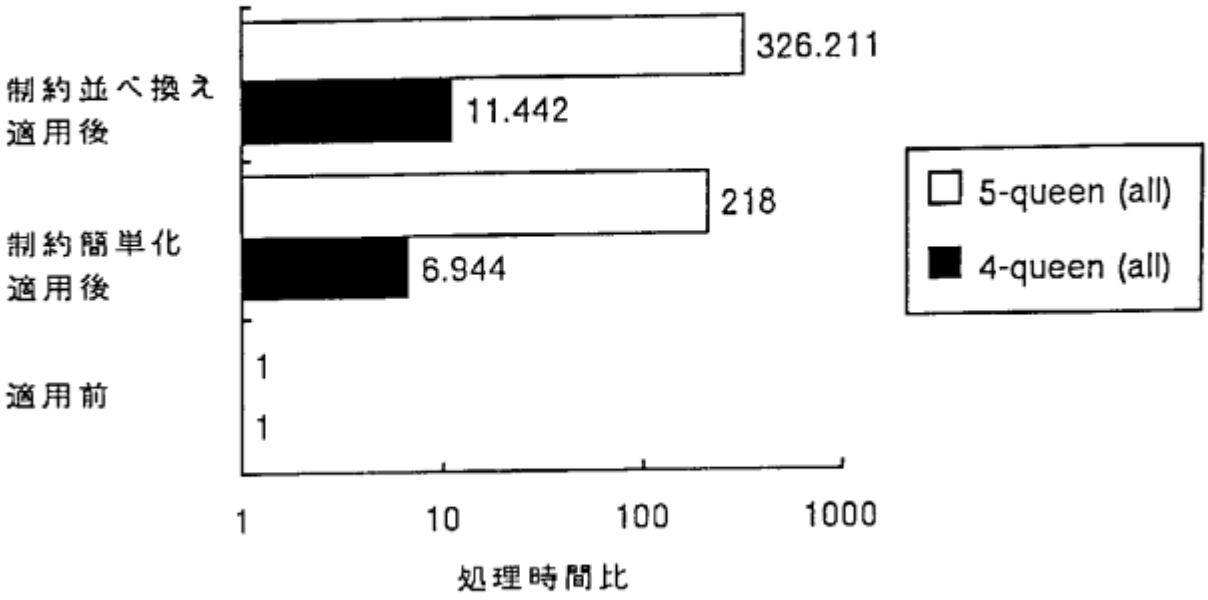

図 9　効率化手法適用による数値解法の処理時間の改善（クイーン問題の全解）
Fig. 9　Performance improvement of n-queen problems (all solutions)