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† 長谷川隆三††

制約充足は人工知能や画像理解の分野をはじめ，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おこなわれている．制約充足問題の代表的な解法として，探索法や整合化手法を用いる方法が知られている．われわれは，このような探索主体のアプローチとは対照的な位置付けにある代数的アプローチについて論じる．本アプローチでは，制約論理型言語の探索機構を利用した制約充足問題に対する研究とは異なり，制約論理型言語におけるブール制約評価系を用いて代数的に制約充足をおこなう．本論文では，ブール代数により制約充足問題を定式化し，得られたブール方程式の求解を制約論理型言語CALにおけるブール制約の評価とみなすことにより，解であるブーリアン・グレブナ基底を求める方法について述べる．さらに，ブール制約評価系を用いた制約充足問題の効率化手法として，1) ブール制約の簡単化方式，2) 制約ネットワークの構造情報に基づいた制約の評価順序の決定方式，について提案する．そして，本効率化手法の有効性を確認するために，ブール制約を用いて記述された問題に対して適用実験をおこなう．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，それとは異なるあらたなブール代数評価系を用いた代数的な方法ならびに効率化手法が有効であることを示す．

Boolean Algebraic Approach to Constraint Satisfaction Problems

YASUO NAGAI † and RYUZO HASEGAWA ††

Constraint satisfaction problems (CSPs) involve finding one or all assignments of the values to the variables such that the all constraints are satisfied. These problems are widely investigated in AI and related areas such as *graph labeling*, *puzzles*, and *machine vision*. In CSPs, the search-based techniques such as backtracking and constraint propagation have been studied as their main approaches. In this paper, we propose a boolean algebraic approach, where the CSPs are converted into boolean equation solving problems and these problems are solved using a boolean constraint solver of the constraint logic programming language. In our approach, we formalize constraint satisfaction problems in terms of boolean algebra and we obtain sets of boolean constraints corresponding to these problems. These boolean constraints are handled incrementally by a boolean constraint solver of constraint logic programming language, CAL. The boolean constraint solver is based on a boolean Gröbner basis computation algorithm, an extension of Buchberger's algorithm. Furthermore, we consider to improve an efficiency of boolean constraint solver through the simplification and structural analysis of sets of the boolean constraints. Finally, this improvement method is applied to some examples showing its effectiveness of the method.

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おこなわれている[8),15)]．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[8),11),15)]，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[8),15)]．

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，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おこなうことができる．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[4)]．このような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，論理型言語の特徴である導出メカニズムに整合化手法を取り入れた探索機構を利用したアプローチ[7),10)]がとられている．

これに対して，本研究では上記のような探索法を用いる制約充足問題の解法とは異なった代数的な解法と

† (株)東芝 研究開発センター システム・ソフトウエア生産技術研究所
Systems and Software Engineering Laboratory, Research Development Center, Toshiba Corporation
†† (財)新世代コンピュータ技術開発機構
Institute for New Generation Computer Technology

して，ブール代数を用いた定式化から得られるブール方程式（これをブール制約という）を求解するアプローチを導入する．探索法を用いる解法によるアプローチでは，生成検査法やバックトラック法などに基づき解が求められ，特に，すべての解を求める場合には，探索木全体の探査が要求される．一方，われわれのアプローチでは，バックトラックなどの探索をおこなわないで，代数方程式の求解により解が求められる．特に，ブーリアン・グレブナ基底を用いる解法によるブール代数方程式の求解では，解の可解性が判定でき，その結果，基底が存在しない場合には解が存在しないことが示される．基底が存在する場合には解の存在が示され，全解をあらわす解を基底形式で求めることができる．

本稿では，ブール代数[16]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．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[14]を説明する．第3章では，具体的な例題を用いて，ブール代数による制約充足問題の定式化を説明する．第4章では，定式化により得られたブール方程式の求解による制約充足処理法の特徴ならびにその効率化手法について述べる．効率化手法としては，ブール制約を簡単化する方法と制約ネットワークの構造情報を利用する方法について説明する．第5章では，まず，制約論理型言語CAL[17]のブール制約評価系のブーリアン・グレブナ基底による解法[18]を用いた制約充足処理法の特徴を説明し，次に，その効率化手法について示す．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[8),15)]．

- 変数集合：$V=\{v_1, \cdots, v_n\}$，各 v_i は互いに独立な変数．
- ドメイン：離散値の有限集合 $D_i=\{d_{i1}, \cdots, d_{im}\}$，$i=1, \cdots, n$，$d_{ij}$ $(j=1, \cdots, m)$ は数値または記号値をあらわす．各変数 v_i には D_i の要素である値 d_{im} が割り当てられる．
- 制約集合：$C=\{c_1, \cdots, c_l\}$．ここで，各 c_i は制約で，それは次のような等式

  $(v_1, v_2, \cdots, v_n)$

  $=$〈直積 $D_1\times D_2\times\cdots\times D_n$ の部分集合〉

  の表現形式をとる（3.1節の例を参照）．

ところで，n 個の変数に対する制約，つまり n 項制約は2項制約により表現できるので[6]，今後は，2項制約のみを対象とした制約充足問題を考える．

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 $\langle B, \wedge, \vee, \neg, 0, 1\rangle$ である[16]．

ブール多項式は，0ならびに1を含まない変数集合 $V=\{v_1, \cdots, v_n\}$ に対して：a) $v_1, \cdots, v_n$，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*．

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 $V=\{v_1, \cdots, v_n\}$，各変数 v_i がとりうる離散値の有限集合を $D_i=\{d_{i1}, \cdots, d_{im}\}$，$(i=1, \cdots, n)$ とする．変数 v_i に対するドメイン D_i の要素 d_{ij} 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，$i=1, \cdots, n$ とする．v_i の値として，d_{ij} $(j=1, \cdots, m)$ のどれかが必ず割り当てられることにより，次式が得られる．

$$\left(\bigvee_{j=1}^{m} x_{ij}\right)=1 \qquad (1)$$

次に，各変数 v_i の値が同時に d_{ij} と d_{ik} $(j\neq k)$ となることはない．これを(2.1)または(2.2)のようなブール方程式として表現する．

$$\bigwedge_{1\leq j, k\leq m, j\neq k} x_{ij}\wedge x_{ik}=0 \qquad (2.1)$$

* 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．

$$\bigwedge_{1\le j,\,k\le m,\,j\neq k} \neg x_{ij}\vee\neg x_{ik}=1 \qquad (2.2)$$

（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．

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

2項タプル集合 T_{ij} 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 の要素を割り当てる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_{ik} と d_{jl} の組み合わせを2項タプル集合 $T_{ij}=\{(d_{ik}, d_{jl})|d_{ik}\in D_i\wedge d_{jl}\in D_j\}$ とし，v_i の値が d_{ik} となり，かつ v_j の値が d_{jl} となることを，以下のブール方程式により表現する．

$$\bigvee_{(d_{ik},\,d_{jl})\in T_{ij}} x_{ik}\wedge x_{jl}=1,\quad j=1,\cdots,m \qquad (4)$$

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_{ik} と d_{jl} からなる組み合わせを2項のタプル集合 $\bar{T}_{ij}=\{(d_{ik}, d_{jl})|D_i\times D_j-T_{ij}\}$ とし*，$\bar{T}_{ij}$ の属する (d_{ik}, d_{jl}) に対しては，以下の(5.1)または(5.2)のようなブール方程式により表現する．

$$\bigwedge_{(d_{ik},\,d_{jl})\in \bar{T}_{ij}} x_{ik}\wedge x_{jl}=0,\quad j=1,\cdots,m \qquad (5.1)$$

$$\bigwedge_{(d_{ik},\,d_{jl})\in \bar{T}_{ij}} \neg x_{ik}\vee\neg x_{jl}=1,\quad j=1,\cdots,m \qquad (5.2)$$

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 具体例による制約充足問題の説明

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_{1j} $(j=1,2)$ を1つだけとるとすると，$X=v_1$，$D_X=D_1$，$d_{11}=5$，$d_{12}=2$ であり，ブール方程式により表現すると $x_{11}\vee x_{12}=1$，$\neg x_{11}\vee\neg x_{12}=1$ が得られる．

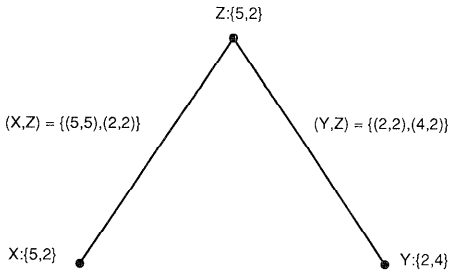


図1 制約ネットワーク1
各頂点には変数とそのドメインが，各辺にはそれが結ぶ2つの頂点間の制約が対応する．

Fig. 1 Example of constraint network 1.
Each vertex corresponds to a variable and its domain. Each edge corresponds to a constraint between two vertices.

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 $T_{13}=\{(d_{11}, d_{31}),(d_{12}, d_{32})\}$，$T_{23}=\{(d_{21}, d_{32}),(d_{22}, d_{32})\}$ 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 $\bar{T}_{13}$, $\bar{T}_{23}$ とすると $\bar{T}_{13}=\{(d_{11}, d_{32}),(d_{12}, d_{31})\}$，$\bar{T}_{23}=\{(d_{21}, d_{31}),(d_{22}, d_{31})\}$ となる．これをブール方程式により表現すると

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

または，

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

* なお，$D_i\times D_j-T_{ij}$ におけるオペレーターは，集合差演算をあらわす．

$x_{21} \wedge x_{31}=0, \quad x_{22} \wedge x_{31}=0$

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

前節の制約充足問題の定式化では，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る．たとえば，後者の場合には次のようなブール方程式集合 $BE1$ が生成される．

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

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2つの充足解

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

$(x_{11}, x_{12}, x_{21}, x_{22}, x_{31}, x_{32})-(0, 1, 1, 0, 0, 1)$

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．

一般に，n 個の要素からなる変数集合，各変数に対応するドメインの要素数が m 個の制約充足問題において，それぞれの局所的な制約関係を探すだけならば，ドメインのサイズにほぼ比例した数の組合せを調べればよい．しかしながら，実際には局所的な制約関係をすべて満足する解の組合せを調べる必要があり，ワーストケースの場合では，$O(m^n)$ 時間を要する．

われわれが提案したブール代数の定式化では，ドメインのサイズにほぼ比例した数の方程式が生成されるが，これらのブール方程式は変数に関する制約および2変数間において成り立つ制約に関する定式化により比較的簡単に得ることが可能である．上記の $BE1$ 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．

また，制約充足問題の特徴を表現する尺度として，制約の充足可能比 R[11]が提案されている．この R は，S/M $(0 \leq R \leq 1)$ で定義される．ただし，S は制約 C をあらわすタプルの数で，$S=|T|(T \subseteq D)$ で定義され，M は C のドメインの直積集合の大きさをあらわし，$M=|D|=\prod_{v_i \in V}|D_i|$ で定義される．探索法を用いた制約充足では，上記の尺度に基づき分類された問題の特徴に適した効率化手法の選択・適用が必要である．たとえば，制約の充足可能比が小さい，つまり制約が緩い問題では，単一解は単純なバックトラックにより容易に求められるが，すべての解を求めたい場合には，より広い探索空間が必要となり，求解は困難になるので，別の効率化手法の検討が必要である．一方，制約の充足可能比が大きい，つまり制約がきつい問題で，単一解を求めるには，単純なバックトラックだけでは多くの無駄な探索処理が必要になり，制約が緩い問題と比較して求解が困難になる．すべての解を求める場合には，単一解を求める場合と比較して探索空間が狭くなるので，別の効率化手法の検討が必要である．われわれが提案する手法では，このような制約充足問題の特徴に適した探索の効率化手法を選択・適用する必要なく，連立方程式の代数解法の効率化手法の適用のみを考慮すればよいという利点がある．

4. ブール方程式解法による制約充足処理の特徴と効率化検討

本章では，ブール方程式解法による制約充足処理の特徴を述べ，その効率化手法について検討する．

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[8),15)]．探索法を用いる解法では，生成検査法やバックトラック法などに基づき解が求められ，すべての解を求める場合には，探索木全体の探査が必要である．これに対して，ブール方程式による解法では，バックトラックなどによる探索をおこなわないで，ブール方程式の求解により解を求めるところに特徴がある．

ブール方程式（以下ではブール制約とよぶことにする）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き分類される．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[21)]．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[3),17),18)]．

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[13)]．しかしながら，代数制約解法の効率化手法は，制約の数と変数の数を比較して，どちらか一方が他方を大きく上回る場合にはあまり有効ではない．その理由は，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，ブール制約の数が一般に取り扱う変数の数と比較して非常に増加する可能性が高くなるからである．このような定式化では，ブール制約の求解に時間を要することになり，効率化手法が必要である．

そこで，ブール制約解法による制約充足処理の効率化手法として，ブール制約の簡単化による手法ならびに制約充足問題のもつ構造情報を用いる手法について述べる．

4.1 ブール制約の簡単化による効率化

ブール制約の簡単化とは，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たのち，乗法標準形をより小さな式に分解することである．ここで，ブール多項式の乗法標準形とは，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$\vee$で結合したもので，いくつかのブール多項式 A_i $(=\vee_{j\in J(i)}B_j)$ (ただし，$J(i)\subseteq\{1, \cdots, n\}$) を論理積演算$\wedge$で結合したもので，$A_1\wedge\cdots\wedge A_n$ なる形のブール多項式としてあらわされる．たとえば，図1の制約ネットワークで表現された問題では，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 $A_1=1, \cdots, A_n-1$ と等価なので，簡単化として $\neg x_{11}\vee\neg x_{32}=1$，$\neg x_{12}\vee\neg x_{31}=1$ が得られる．この手法では，**図2**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ができるだけ簡単な制約に変換される．ここでは，ブール制約の簡単化を容易にするために，2項制約を禁止条件である割り当てを許さない表現を用いて定式化する．上述の図1の制約ネットワーク問題の定式化は，2項制約を禁止条件として表現したものである．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．

4.2 制約ネットワークの構造情報を用いた効率化

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，木探索法でのバックトラック処理を改善する手法や併合法が検討されている．これらの効率化手法の多くは，2個の変数間の制約（2項制約）集合により表現された問題，すなわち制約ネットワークのグラフ表現において，頂点とエッジがそれぞれ変数と制約に対応付けられている問題のみを対象として検討されてきた．しかしながら，3個以上の変数により表現された制約（n項制約）からなる問題に対しても，頂点を制約に，エッジを変数にそれぞれ対応付けてグラフ表現*することにより，構造情報を得ることは可能である．したがって，一般に，制約集合に対しては常に制約ネットワークが存在し，その構造情報が得られる．

併合法[19]では，このような制約ネットワークのグラフ表現においてエッジの両端の頂点を1個の新しい頂点にまとめる併合操作を繰り返し，最終的に頂点が1個だけのネットワークに縮退する．その結果，頂点の併合ならびに除去されるエッジの順序が決定される．このような併合操作により，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が繰り返される．

われわれは，併合法と同様に制約ネットワークの変更操作により決定される頂点の順序に基づいた効率化手法を提案する．本効率化手法では，**図3**に示されるような処理をおこなう．具体的には，まず，制約ネットワークのグラフ表現 $G=(V, E)$ に対して最も小さい次数をもった頂点を選択する．なお，次数とは各頂点に接続するエッジの数をあらわす．ただし，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の充足可能比 R を最も小さくする（制約をきつくする）頂点を選択する．次に，選択された頂点 $v\in V$ とそれに接続するすべてのエッジ E_v をグラフ $G=(V, E)$ から取り去ることで部分グラフ $G'=(V-\{v\}, E-E_v)$ が得られる．さら

```
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 =
13      ¬A = 1:
14         Constraint' ← Constraint' ∪ {A = 0};
15      ¬A = 0:
16         Constraint' ← Constraint' ∪ {A − 1};
17   end.
```

図2 ブール制約の簡単化アルゴリズム
ここで，$A_1\wedge\cdots\wedge A_n=1$（3行目）は乗法標準形式のブール制約をあらわし，各 A_i はリテラルを論理和演算で結合したブール多項式をあらわす．$\neg A_1\vee\cdots\vee\neg A_n=1$（9行目）ならびに$\neg A_1\vee\cdots\vee\neg A_n=0$(11行目)は，負リテラルからなる加法標準形式のブール制約をあらわす．$\neg A=1$(13行目）および$\neg A-0$(15行目）は，負リテラルからなるブール制約をあらわす．

Fig. 2 Transformation algorithm of boolean constraints. $A_1\wedge\cdots\wedge A_n=1$ (line 3) describes a conjunctive normal form, where A_i describes a boolean formulas connecting literals with an operator of logical addition. $\neg A_1\vee\cdots\vee\neg A_n=1$ (line 9) and $\neg A_1\vee\cdots\vee\neg A_n=0$ (line 11) describe boolean constraints consisting of negative literals. $\neg A=1$ (line 13) and $\neg A=0$ (linc 15) describe boolean constraints including a negative literal.

* 実際には，制約ネットワークはハイパーグラフ[2]として表現される．

```
:- 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.
```

図3 制約ネットワーク問題を記述したCALプログラム（プログラム①）

Fig. 3 CAL program describing constraint network problem (program ①).

に，このような変形操作を順番に繰り返すことにより，最終的には，頂点が1個だけのグラフが得られる．以上のようにして，選択された変数の順序に基づき，グラフを変形することで除去される頂点とエッジに対応する制約を並び換えることで，制約集合の求解の効率化をはかる．

5. 制約論理型言語のブール制約評価系

5.1 ブーリアン・グレブナ基底を求めるブール制約評価系

制約論理型言語は，論理型言語(たとえば，Prolog)を基本言語として，ある計算領域における制約に関する記述能力および制約充足能力を組み込んだ言語である[4),17)]．その構造は全体の処理を制御する推論機構と制約を解く制約評価系からなる．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る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[17),18)]．

ブール制約評価系は，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[3)]に基づき項書き換えをおこない，ブーリアン・グレブナ基底を求める．これは項書き換え処理とみなせ，停止性ならびに合流性という性質をもつ．なお，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[16)]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を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[18)]．

5.2 ブーリアン・グレブナ基底を求めるブール制約評価系を用いた制約充足解法の特徴とその効率化検討

ブール制約の求解をおこなう制約論理型言語CALのブール制約評価系を用いた制約充足解法の特徴とその効率化について検討する．

先ほど述べたように，ブール方程式解法による制約充足処理では，バックトラックなどによる探索をおこなわないで，代数方程式の求解により解を求めるところに特徴がある．さらに，ブーリアン・グレブナ基底を用いる解法の特徴から，ブール代数方程式の求解において，基底が存在しない場合には解が存在しないことが判定できる．また，基底が存在する場合には解の存在が判定でき，解空間を基底形式で求めることができる．なお，求められた基底は解空間を表現することになるので，われわれの提案した解法では健全性ならびに完全性が保証されることになる．

次に，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，頂点の次数および制約の充足可能比 R により決定される．

ブール制約の簡単化では，以下を考慮した処理がおこなわれる．ブール代数 $\langle B, \wedge, \vee, \neg, 0, 1\rangle$ は単位元をもつブール環 $\langle R, +, \cdot, 0, 1\rangle$ と同等であるとみなされる[16)]．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，演算 $+, \cdot$ において $x+x=0$ 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 $\langle R, +, \cdot, 0, 1\rangle$ 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[16)]．このようなブール演算 $\wedge, \vee, \neg$ により表現されたブール制約が複雑であれば，制約評価において用いられるブール環上の多項式も一層複雑な形式をとり，ブーリアン・グレブナ基底を求める処理に時間を要する．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，ブール制約が簡単な形式で与えられることが望ましい．効率化手法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．

さらに，制約の評価順序による効率化では，4.2節の制約ネットワークの構造情報を用いる手法に加えて，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方法が有効であると考えられる．その理由は，たとえば，論理和演算からなる多項式である $x \vee y$ と $x \vee y \vee z$ がそれぞれ一層複雑なブール環上の多項式 $x+y+x \cdot y$ と $x+y+x \cdot y+z+(x+y+x \cdot y) \cdot z$ に変換されるからである．したがって，ブール制約評価系では，論理和演算∨を含んだ制約はブール環上の多項式に変換された後でグレブナ基底が求められ，一層複雑な基底が求められる可能性があるので，なるべくあとで評価するようにする．

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，いくつかの例題に対して定式化をおこない，制約論理型言語CALのブール制約を用いてプログラムを記述する．なお，ここでは，変数に関する制約をブール制約の簡単化の容易性を考慮して禁止制約として表現する．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[17]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．

まず，制約ネットワークにより表現された例題を，次のようなCALで記述したプログラムを図3のプログラム①とする．なお，bool：X11\/X12=1はブール制約をあらわし，\/は論理和∨を，& は論理積∧を，~X11はブール変数X11の否定をそれぞれあらわす[20]．

プログラム①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，$\bar{x}_{21}$ はブール変数 x_{21} の否定形をあらわす．

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

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，この例に対して効率化手法を適用する．はじめに，ブール制約効率化手法として，簡単化処理がおこなわれたプログラムを図4のプログラム②とする．簡単化処理では，まず，プログラム②中のブール制約，たとえばbool：~X11\/~X12=1は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_networkn/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.
```

図4 制約簡単化処理適用により得られたCALプログラム（プログラム②）

Fig. 4 CAL program generated by applying constraint simplification processing (program ②).

そして，制約ネットワークの構造情報を用いた効率化手法により頂点の選択順序が，図5のアルゴリズムに基づき，以下のように決定される．まず，図1の制約ネットワークのグラフ表現 $G=(V, E)$ において各頂点 $v \in V$ の次数を求めると頂点 x が1，y が1，z が2となる．次に，これらの頂点のなかで，最小次数の頂点を選択頂点とみなすことにする．この場合は，該当する頂点は x と y が存在するので，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．図1の例では，2項制約 C_{xz} と C_{yz} の充足可能比 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．このように複数の頂点候補が存在する場合には，$x>y>z>\cdots$ のような辞書式順序[22]に基づき x が頂点として選択される．さらに，$G=(V, E)$ から選択頂点 x とそれに接続するすべてのエッジ E_x を取り除くことで，部分グラフ $G'=(V-\{x\}, E-E_x)$ が得られる．得られた部分グラフ G' から同様な操作により頂点 y を選択し，部分グラフ $G''=(V'-\{y\}, E'-E'_y)$ が得られる．最終的には1個だけの頂点 z からなるグラフが得られる．その結果，$x \Rightarrow y \Rightarrow z$ という頂点の選択順序が求められる．このように決定された頂点の選択順序およびエッジの除去順序に基づき，並べ換えられたブール制約をボディー部のサブゴールに

```
procedure vertex_select(G(V, E): input, Vertices: output)
1    G_0 ← G(V, E);
2    i ← 0;
3    Vertices ← 0;
4    R_min ← ∞;
5    while G_i ≠ 0 do
6      DV ← {v | degree of v is minimum such that v ∈ V};
7      for every dv ∈ DV
8        Adj_dv ← {w | (dv, w) ∈ E such that w ∈ V};
9        R_dv ← Σ_{j∈Adj_dv} R_dv,j;
10       R_min ← min(R_dv, R_min);
11     endfor
12     Pick a vertex dv such that R_dv = R_min and dv ∈ DV;
13     v_i ← dv;
14     Vertices ← Vertices ∪ {v_i};
15     G_{i+1} ← G(V − {v_i}, E(V − {v_i}));
16     i ← i + 1;
17   endwhile
18   end.
```

図5 頂点選択アルゴリズム
1行目における $G(V, E)$ は，制約集合に対応する制約ネットワークのグラフ表現をあらわす．6行目における DV は頂点集合 V のなかで，最小次数をもつ頂点集合をあらわす．8行目における Adj_{dv} は最小次数の頂点 dv に隣接する頂点集合をあらわす．9行目における R_{dv} は，頂点 dv と頂点集合 Adj_{dv} の要素 j 間の制約の充足可能比 $R_{dv,j}$ の総和をあらわす．12行目における d_v は，頂点 d_v に対応した制約の充足可能比の総和 R_{dv} のなかで，最小値 R_{min} をもつ頂点をあらわす．14行目における $Vertices$ は選択頂点集合をあらわす．15行目における $G(V-\{v_i\}, E(V-\{v_i\}))$ は，グラフ $G(V, E)$ から選択された頂点 v_i とそれに接続するすべてのエッジの除去により得られる部分グラフをあらわす．

Fig. 5 Vertex selection algorithm.
$G(V, E)$ (line 1) describes graphical representation of a constraint network corresponding to the constraint set. DV (line 6) describes a subset of V having a minimum degree. Adj_{dv} (line 8) describes the set of vertices adjacent to the vertex with minimum degree. R_{dv} (line 9) describes summation of satisfiability $R_{dv,j}$ between a vertex dv and an element j in the vertex set Adj_{dv}. d_v (line 12) describes the selected vertex with the same value of constraint satisfiability R_{dv} corresponding to the vertex dv as the minimum value R_{min}. v_i (line 13) describes d_v and $Vertices$ (line 14) describes a set of the selected vertice. $G(V-\{v_i\}, E(V-\{v_i\}))$ (line 15) describes a subgraph reduced by removing the selected vertex v_i and all edges incident to this vertex from the graph $G(V, E)$.

もつプログラムを**図6**のプログラム③とする．

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 $1 \leq i \leq n$ 列に対応し，それぞれの変数 v_i のドメインが $D_i=\{1, \cdots, n\}$ であり，変数の値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．

```
:- public c_networknn/6.

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

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

図6 制約簡単化および制約並べ換えにより得られた CAL プログラム（プログラム③）

Fig. 6 CAL program generated by applying constraint simplification and ordering (program ③).

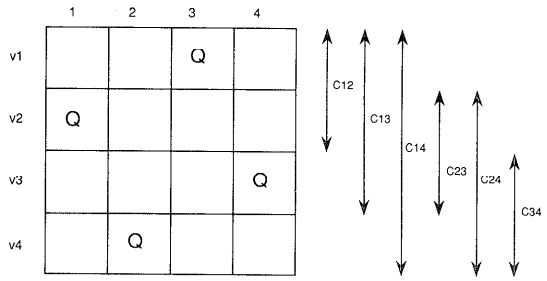


図7 2項制約による4クイーン問題の定式化
ここで，C_{ij} は i 番目と j 番目 (i=j) のクイーン間の配置に関する制約をあらわす．

Fig. 7 Formalization of 4-queen problem using binary constraint.
C_{ij} expresses the binary constraints so that i-th and j-th queens don't attack each other (i=j).

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[11]．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 $(v_i \neq v_j) \wedge (|v_i - v_j| \neq j \quad i)$ (ただし，$1 \leq i < j \leq n$) 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7**のように6個の2項制約として表現され[11]，制約ネットワークは，**図8**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し，たとえば，ノード v1 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 v_1 の値として，ドメイン D_1 からどれかひとつの要素 d_{1_k} を必ず割り当てる制約をあらわし，C_1 は変数 v_1 の値が同時に d_{1_j} と d_{1_k} $(j \neq k$ かつ $1 \leq j, k \leq n)$ をとらない制約をあらわす．ノード v1 と v2 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

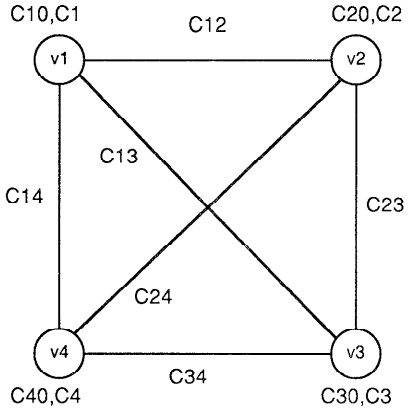


図8 制約ネットワーク（4クイーン問題）
ここでは，頂点には頂点に対応する変数に関連する制約のみを，各辺には頂点に対応する2変数間の制約を割り当てる．

Fig. 8 Constraint network representation corresponding to 4-queen problem.
Vertex and edge respectively correspond to a problem variable and binary constraint between two variables.

C_{12} に対応する．図8のように表現された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CALで記述されたプログラムとして実行される．

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，上記のCALプログラムに対して，ブール制約の簡単化をおこなう．さらに，制約集合の構造情報を用いて決定された頂点の選択順序に基づき，この簡単化されたプログラムの制約の並べ換えをおこなう．頂点の選択順序は図5のアルゴリズムにより決定される．まず，図8の制約ネットワークから，各頂点 v_1, v_2, v_3, v_4 の次数(この場合は完全グラフ K_4 として表現されるので，すべての次数が3となる)が求められる．そして，各々の頂点 v_i 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 R_v が最小となる頂点を選択する．なお，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．ここでの選択候補の頂点は v_2 と v_3

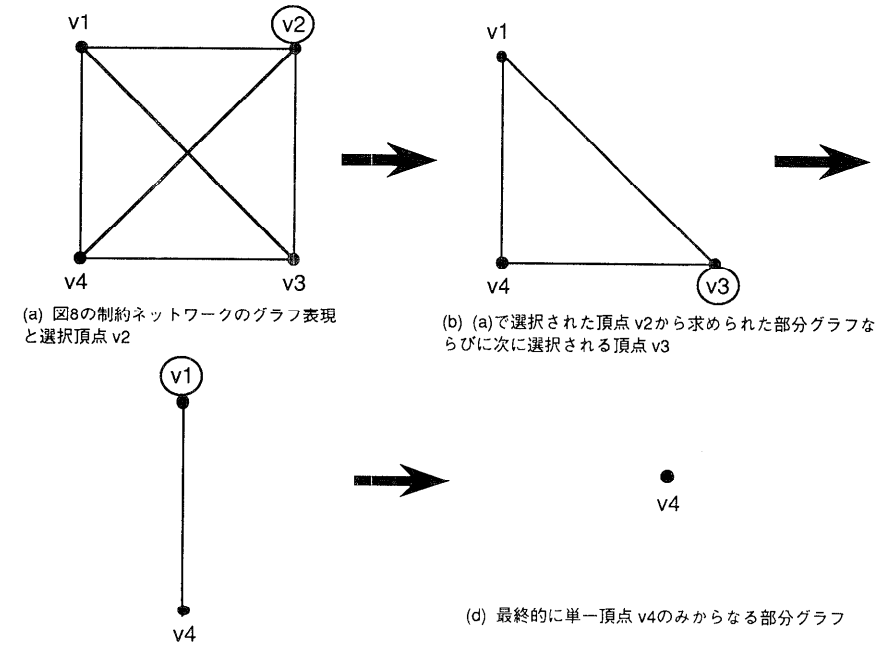


図9 図8の制約ネットワークのグラフ表現から求められる頂点の選択順序

Fig. 9 Vertex selection process determined from a graphical representation of constraint network as shown in Fig. 8.
(a) Graphical representation of constraint network as shown in Fig. 8 and the selected vertex v2.
(b) A result of graph reduction using the vertex v2 and the next selected vertex v3.
(c) A result of graph reduction using the vertex v3 and the next selected vertex v1.
(d) Final result of graph reduction showing the single vertex v4.

であるが，制約の充足可能比の和が最小となる頂点 v_2 が選択される．そして，制約ネットワークのグラフ表現から選択頂点 v_2 とそれに接続するすべてのエッジを取り除くことにより得られた部分グラフに対して，さらに上記の操作を繰り返す．最終的に，**図9**のように $v_2 \Rightarrow v_3 \Rightarrow v_1 \Rightarrow v_4$ という頂点の選択順序が決定される．

また，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，5.2節に示されたように制約評価においてブール制約が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ことになる．したがって，プログラムのボディ部では，制約評価により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後で評価するようにゴールの順序を決定する．

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10**から**図12**に示される．実験は逐次推論マシンPSI-II[12]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，制約の数が多い場合にはその処理が非効率になる可能性が高い．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と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適用した．

ブール制約評価系の効率化手法として，まず，ブール代数による定式化で得られたブール制約集合が記述されたCALプログラムを実行する処理を「効率化手法の適用前」と呼ぶ．次に，否定ならびに論理和演算を含まないようにブール制約を簡単化したプログラムを実行する処理を「制約簡単化」と呼ぶ．**表2**は各問題に対する「制約簡単化」の適用結果を示す．さらに，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

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 constraints.

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

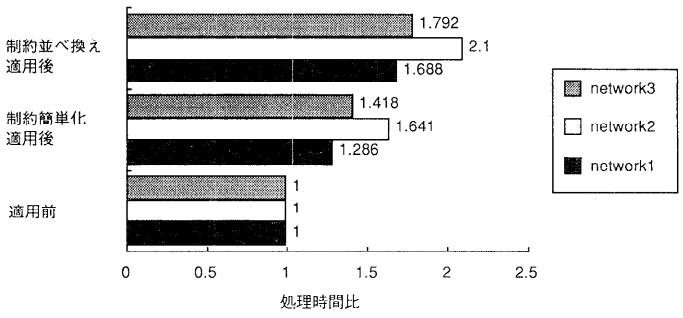


図10 効率化手法の適用による処理時間の改善（制約ネットワーク問題）
Fig. 10 Performance improvement of constraint network problems.

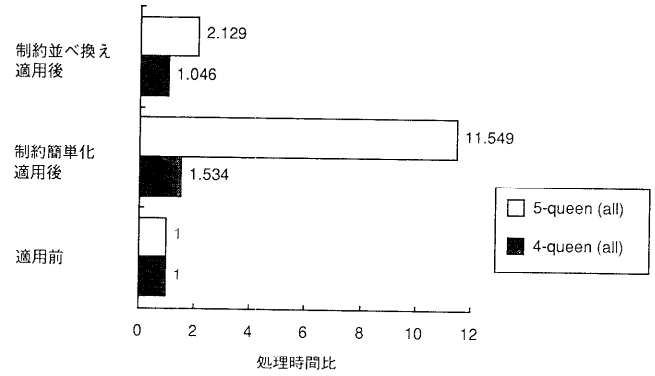


図 11 効率化手法適用による記号解法の処理時間の改善（クイーン問題の全解）
Fig. 11 Performance improvement of n-queen problems (all solutions).

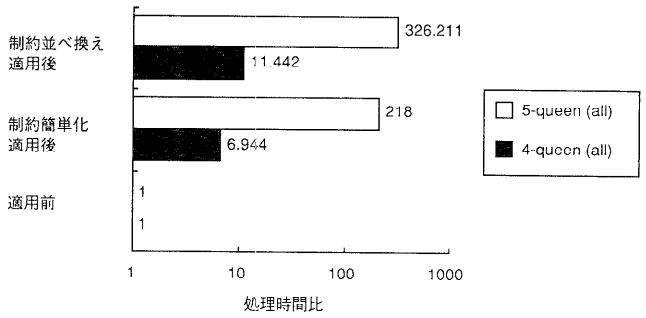


図 12 効率化手法適用による数値割り当て解法の処理時間の改善（クイーン問題の全解）
Fig. 12 Performance improvement of n-queen problems (all solutions).

価順序を修正し，プログラムを実行する処理を「制約並べ換え」と呼ぶ．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 10 から図 12 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ている．

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「制約並べ換え」を適用した「記号による解法」では，約 69%から 2.1 倍の処理時間の改善がみられ有効であることがわかった（**図 10**）．

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 53%の処理時間の改善がみられたが，「制約並べ換え」においては改善がほとんどみられなかった(**図 11**)．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，「数値割り当てによる解法」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 11)．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 2.1 倍の改善にとどまった（図 11)．

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「制約並べ換え」を適用した「数値割り当てによる解法」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「数値割り当てによる解法」では，「制約簡単化」により約 6.9 倍の処理時間の改善がみられ，「制約並べ換え」により約 11.4 倍の処理時間の改善がみられた(**図 12**)．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「数値割り当てによる解法」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，「制約並べ換え」により約 326 倍もの処理時間の改善がみられた(図 12)．

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のに対して，ブーリアン・グレブナ基底を求める手法で得られる解は，ブール変数間の関係をあらわす基底形式をとり，列挙解とはなっていない．そのため，探索法による列挙解を得るためには，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が解として求められる．一方，「数値割り当てによる解法」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，簡単化可能な制約ができる限り簡単化されたグレブナ基底が求められる．この解法は，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「制約並べ換え」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，$P_1 = 1, \cdots, P_n = 1$ のように多くの制約が生成される．さらに，「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・オフを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の充足可能比に基づき，頂点の選択順序が決定される．ここでは，探索処理の効率化手法で用いられた制約の充足可能比 R という尺度が使われ，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エッジに対応する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「制約並べ換え」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，「制約並べ換え」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クイーンおよび5クイーン問題に対して「制約簡単化」ならびに「制約並べ換え」を適用した「数値割り当てによる解法」では，処理時間のおおきな改善がみられた．その理由は「数値割り当てによる解法」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クイーンおよび5クイーン問題に対して「制約簡単化」ならびに「制約並べ換え」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「制約並べ換え」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増加や項の複雑化を抑制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[7]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[7]．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，「制約並べ換え」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いえる．本手法では，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，制約が静的に与えられている場合と同様に対処できる．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，制約の除去などの処理には不十分であり，詳細な検討が必要である．また，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．

従来，制約充足問題の解法として生成検査法やバックトラック法などを用いた探索による解法がよく知られているが，これらの解法によりすべての解を求める場合には，探索木全体の探査が必要である．これに対して，われわれの提案したアプローチではバックトラックなどの探索をおこなわないで，ブーリアン・グレブナ基底を用いてブール方程式の解を求める．その結果，次のように解の可解性を判定できるという特徴を有している．

- 基底が存在しない場合には解が存在しないことが示される．
- 基底が存在する場合には解の存在が示され，全解をあらわす解を基底形式で求められる．

また，制約充足問題では，タプル集合のように外延的に (extensionally) 表現される制約と論理式のように内包的に (intensionally) 表現された制約が取り扱われているが，現状では前者の外延的な制約のみが取り扱われている場合が多い．したがって，両者の表現をいかに統一的に取り扱うかが問題となっている[15]．このような問題に対して，われわれの提案したブーリアン・グレブナ基底による解空間の表現形式を利用することで，両者の制約表現を統一的に扱うことが容易になると思われる．

本論文で示したような解空間基底を用いたアプローチの有望な応用としては，関係データベースにおける一貫性情報のインクリメンタルな管理[23]や知識ベースにおけるインクリメンタルな真偽維持[5),9)]が考えられる．

なお，本論文で提案した効率化手法は，現状では他の制約充足手法と比較して計算効率が劣っているが，グレブナ基底計算アルゴリズムの効率化や並列化による高速化に関する研究[24),25)]がおこなわれており，将来的にはこれらの結果を適用することで計算効率の改善が期待できる．

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おこなわれた．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（現東京大学教授），古川康一研究次長（現慶応義塾大学教授），新田克己第7研究室室長（現第2研究部長），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（現第2研究部長代理）ならびにNTTデータ通信(株)の生駒憲治氏（元ICOT研究部長代理）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ICOT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・ソフトウエア生産技術研究所の西島誠一所長（現東芝ドキュメンツ），河野毅部長（現東京システムセンター）ならびに渡辺貞一現所長に深謝いたします．

参考文献

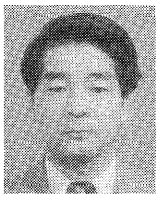
1) Beineke, L. W. and Wilson, R. J. (eds.): *Selected Topics in Graph Theory 2*, Academic Press (1983).
2) Berge, C.: *Graphs and Hypergraphs*, North-Holland (1973).
3) Buchberger, B.: Gröbner Bases: An Algorithmic Method in Polynomial Ideal Theory, *Multidimensional Systems Theory* (Bose, N. ed.), pp. 184-232, D. Reidel Publ. Comp., Dordrecht (1985).
4) Cohen, J.: Constraint Logic Programming Languages, *Comms. ACM*, Vol. 33, No. 7, pp. 52-68 (1990).
5) de Kleer, J.: A Comparison of ATMS and CSP Techniques, *Proc. of the IJCAI-89*, pp. 290-296 (1989).
6) Freuder, E. C.: Synthesizing Constraint Expressions, *Comms. ACM*, Vol. 21, No. 11, pp. 958-966 (1978).
7) Hentenryck, P. V.: *Constraint Satisfaction in Logic Programming*, MIT Press (1989).
8) Mackworth, A. K.: Constraint Satisfaction, *Encyclopedia of Artificial Intelligence* (Shapiro, S. C. ed.), Vol. 1, pp. 205-211, John Wiley & Sons, Inc. (1987).
9) McAllester, D.: Truth Maintenance, *Proc. the AAAI-90*, pp. 1109-1116 (1990).
10) Montanari, U. and Rossi, F.: Perfect Relaxation in Constraint Logic Programming, *Proc. ICLP '91*, pp. 223-237 (1991).
11) Nadel, B. A.: Representation Selection for Constraint Satisfaction: A Case Study Using n-Queens, *Proc. IEEE Expert*, pp. 16-23 (1990).
12) Nakashima, H. and Nakajima, K.: Hardware Architecture of the Sequential Inference Machine: PSI-II, *Proc. 1987 Symposium on Logic Programming*, pp. 104-113 (1987).
13) 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).

14) 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，AI 90-73 (1991).
15) 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).
16) Rudeanu, S.: *Boolean Functions and Equations*, North-Holland Publ. Comp. (1974).
17) Sakai, K. and Aiba, A.: CAL: Theoretical Background of Constraint Logic Programming and its Application, *J. Symbolic Computation*, Vol. 8, pp. 589-603 (1989).
18) Sakai, K. and Sato, Y.: Application of Ideal Theory to Boolean Constraint Solving, *Proc. PRICAI '90*, pp. 490-495 (1990).
19) 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).
20) (財)新世代コンピュータ技術開発機構：CAL 言語仕様第1.0版，(財)新世代コンピュータ技術開発機構 (1990).
21) 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).
22) 井田哲雄：岩波講座 ソフトウエア科学12，計算モデルの基礎理論，岩波書店 (1991).
23) Gyssens, M., Jeavons, P. G. and Cohen, D. A.: Decomposing Constraint Satisfaction Problems Using Database Techniques, *Artif. Intell.*, Vol. 66, No. 1, pp. 57-89 (1994).
24) Terasaki, S., Hawley, D., Sawada, H., Satoh, K., Menju, S., Kawagishi, S., Iwayama, N. and Aiba, A.: Parallel Constraint Logic Programming Language GDCC and its Parallel Constraint Solvers, *Proc. the Int'l Conf. on FGCS 1992*, pp. 330-346 (1992).
25) Senechaud, P.: Implementation of a Parallel Algorithm to Compute a Gröbner Basis on Boolean Polynomials, *Computer Algebra and Parallelism* (Dora, J. D. and Fitch, J. eds.), pp. 159-166, Academic Press (1990).

（平成5年4月1日受付）
（平成7年1月12日採録）

永井 保夫（正会員）

1961年生．1983年早稲田大学理工学部電気工学科卒業．1985年同大学大学院理工学研究科電気工学専攻修士課程修了．同年(株)東芝に入社．同年より1989年まで(財)新世代コンピュータ技術開発機構へ出向．現在，同社研究開発センターシステム・ソフトウエア生産技術研究所に勤務．VLSI自動合成システム，エキスパートシステム構築支援ツール，機械設計エキスパートシステム，制約充足問題，制約プログラミングを用いたシステム構築に関する研究に従事．計画型エキスパートシステム，OR，オブジェクト指向プログラミング，GUI構築支援システムなどにも興味をもつ．1989年人工知能学会全国大会優秀論文賞，1991年情報処理学会奨励賞受賞．電子情報通信学会，人工知能学会，日本ソフトウエア科学会各会員．



長谷川隆三（正会員）

1949年生．1972年九州大学工学部通信工学科卒業．1974年九州大学大学院工学研究科通信工学専攻修士課程修了．同年日本電信電話公社入社．同社武蔵野電気通信研究所勤務．1987年(財)新世代コンピュータ技術開発機構へ出向，現在に至る．九州大学博士(工学)．ポリプロセッサシステム，データフローマシン，関数型言語，論理プログラミングおよび定理証明に関する研究に従事．電子情報通信学会会員．