整合ラベリングのための改良拘束伝播法†

西原清一†† 原智亨††† 池田克夫††

複数個の構成要素からなるある対象物を解析する問題においては，まず，各要素に関する局所的に解釈可能なラベル集合を定め，それらのなかから対象物全体の矛盾のない解釈を与えるようなラベルの組みを探索するという方法がある．整合ラベリング (consistent labeling, CL) 問題は，このような問題への一般的な解法を目的とするものであり，線画理解，シーンのラベルづけ，Nクイーン問題などのパズル，さらにグラフの同型写像の探索など多方面の問題への応用が考えられる．CL 問題は，問題対象の構成要素（ユニット）の集合，その解釈の候補（ラベル）の集合，および要素間に成り立つべき解釈の拘束条件の三つによって記述できる．CL 問題への一つの接近方法として，拘束伝播によって可能な解釈の候補を徐々に絞ってゆく方法がある．本稿では，拘束条件が2項関係の集合で与えられたとき，おのおののユニットのラベル集合を2項関係をもとに絞ってゆくアルゴリズム，いわゆる辺整合アルゴリズムについて，従来の方法を考察する．また2項関係においてラベルの出現する回数，すなわちラベル重複度を導入したアルゴリズムを新たに提案し，計算機実験によりその有効性を評価する．

1. まえがき

線画理解（文献1）など）や同型な部分グラフを探す問題（文献2）など）にみられるように，問題の対象が複数の構成要素およびそれらの要素間に成立すべき拘束条件によって記述できるとき，この記述をもとにその対象を解釈したり，与えられたデータのなかからその記述と一致するものを探し出したりする問題は多い．これらは，まず各要素に関する局所的に解釈可能なラベル集合を定め，それらのなかから所与の拘束条件を満足し対象物全体の矛盾のない解釈を与えるようなラベルの組を選別する作業であり，'整合ラベリング (consistent labeling) 問題'[3),4)]，'拘束充足 (constraint satisfaction) 問題'[6)]，'関係整合 (relational consistency) 問題'[7)] などと呼ばれ，パターン認識や人工知能の分野において多く見いだされる．この問題を以後，文献[3),4)]にならって，CL 問題と略記する．CL 問題は NP 完全な探索問題の一種であり，その解決法としては，バックトラッキングを基本とした深さ優先木探索 (tree search) による方法と，弛緩操作 (relaxation)[8)] や拘束伝播 (constraint propagation)[9)] を用いたフィルタリングによる方法の二つに大別される．前者の方法としては，look-ahead オペレータ[3)] や backmarking 技法[10)] を木探索の処理過程に導入し，むだなバックトラッキングのくり返し (thrashing) を未然に防止する方法など種々，提案されている[11)]．一方，後者の方法としては，離散的弛緩法[8)] や二つの構成要素間の拘束伝播による方法[1),5)-7),12)] などがある．本稿は，Mackworth[6)] によって与えられた拘束伝播の方法およびその改良である McGregor[7)] の方法を検討したのち，新たに後者を改良した方法を提案し，実験により各方法の効率の比較を行うものである．すなわち，本稿で与える方法は新たに相互拘束伝播の手法を導入しており，さらに拘束伝播によって削除されることが予想されるラベル（解釈）を陽に同定し，処理の効率化を図っている．

以下，まず2章では CL 問題の定義を与え，既存のアルゴリズムを検討したのち，3章において新たに拘束伝播の改良アルゴリズムを提案する．4章では，実験により諸方法を比較する．

2. 整合ラベリング問題

2.1 問題の定義

ここでは，整合ラベリング問題（CL 問題）を定義する．まず，'ユニット' の有限集合を $U=\{1,\cdots,n\}$ とする．各ユニット $i(1\leq i\leq n)$ に対して 'ラベル' の有限集合 L_i が存在する．ここに，ユニットは，前節で述べたように，問題の対象を分解したときの各構成要素に対応するものである．ラベルはユニットに対する可能な解釈の候補を表す．また，ユニット間に成り立つべき '拘束条件' は2項関係 $R_{ij}\subseteq L_i\times L_j$ の有限集合で与えられる[6),7)]．2項関係の個数を m とする．

† An Improved Constraint Propagation Method for Consistent Labeling by SEIICHI NISHIHARA (Institute of Information Sciences and Electronics, University of Tsukuba), TOSHIAKI HARA (Department of Scientific Technology, University of Tsukuba) and KATSUO IKEDA (Institute of Information Sciences and Electronics, University of Tsukuba).

†† 筑波大学電子・情報工学系
††† 筑波大学大学院理工学研究科

ここで，R_{ij} は二つのユニット i,j に与えるラベルの組に課せられる2項関係を表す．また R_{ij} に対して対称な関係 R_{ji} も存在するものとしておく．とくに二つのユニット i,j に与えるべきラベルに拘束のない場合は，上の定義を自然に拡張して，$R_{ij}=L_i\times L_j$ と考える．ラベルの組 (x,y) が関係 R_{ij} を満たす（すなわち $(x,y)\in R_{ij}$）とき，これを述語表現して，$R_{ij}(x,y)$ が真であるということもある．CL 問題とは全ユニット $(1,\cdots,n)$ に対するラベルの組 $(a_1,\cdots,a_n)$ のうち，すべての2項関係 R_{ij} について $R_{ij}(a_i,a_j)$ が真となっているようなものをすべて見つける問題である．ただし，$a_i\in L_i, 1\leq i\leq n$．また，そのようなラベルの組を‘整合ラベル組’[3]という．一般の CL 問題は NP 完全であり[3),4),7)]，深さ優先探索法などにおいてしばしば見られる組合せ的爆発現象を緩和するための技法の開発は意義のあることである．

‘拘束グラフ’は，CL 問題を無向グラフ $G=(U,E)$ で表現したものである．ここに頂点集合 U は，上のユニット集合である．各頂点には，先に述べたようにラベル集合が一つずつ対応している．辺集合は，$E=\{(i,j)\mid R_{ij}$ が定義されている$\}$ で与える．すなわち各辺にはそれぞれ一つの2項関係が対応している．辺の個数は2項関係の個数 m に等しい．拘束グラフの簡単な例を**図1**に示す．頂点 i から j への‘拘束伝播’prop(i,j) とは，j のラベル集合 L_j の要素のうち関係 R_{ij} を満たさないようなラベルを削除する処理のことである．すなわち，L_j のかわりに新たにつぎのような L_j' をユニット j のラベル集合とする操作である．

$$L_j'=\{y\mid y\in L_j\wedge\exists x[x\in L_i\wedge R_{ij}(x,y)]\}$$

拘束伝播は方向性をもった操作であり，prop(i,j) と prop(j,i) は相異なる処理であることに注意．prop(i,j) において，頂点 i を‘起点頂点’という．図1において，prop$(1,2)$ を施すと，頂点2のラベル集合は新たに $\{a,b,d\}$ に絞られる．ある辺 (i,j) について，prop(i,j) および prop(j,i) を行ってもラベル集合 L_i,L_j がそれ以上小さくならない場合，辺 (i,j) は‘辺整合 (arc consistent)’[6),7)]であるという．すべての辺が辺整合であるような拘束グラフは，‘既弛緩 (relaxed)’であるという．図1の拘束グラフで表された CL 問題の解，すなわちユニットの組$(1,2,3)$に対する整合ラベルの組は，(x,d,b) および (z,d,c) の二つである．整合ラベル組を構成するラベルは，既弛緩状態にある拘束グラフのラベル集合に必ず含まれている．とくに整合ラベル組がただ一つの場合は，既弛緩後の各ラベル集合にはちょうど一つのラベルが残る．またこの逆も成り立つ．したがって，与えられた拘束グラフを既弛緩の状態に変換する手法の開発は重要で，これを辺整合アルゴリズムという．次節では，二つの辺整合アルゴリズムについて考察する．

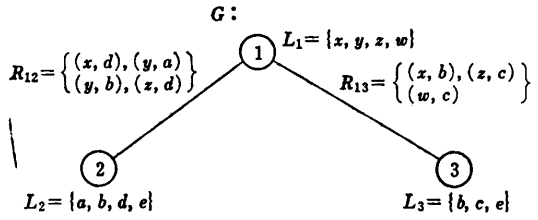

図1 拘束グラフ G の例
Fig. 1 An example of constraint graph G.

2.2 辺整合アルゴリズム

最も基本的な辺整合アルゴリズムとして，Mackworth によって与えられた AC 1 がある[6]．**図2**は，McGregor によって整理された AC 1 アルゴリズムである[7]（以後，これをアルゴリズムAと記す）．これは，拘束グラフ G の各頂点について，そのすべての隣接頂点を起点として当該の頂点への拘束伝播を行うという処理をくり返すものである．このくり返しは，どのラベル集合も変化しなくなるまで行われる．McGregor[7]はこれを改良して**図3**のようなアルゴリズム（これをアルゴリズムBと記す）を与えた．これは，ある一つの拘束伝播 prop(j,i) を行うことによってラベル集合 L_i に変化が生じた場合，すなわち少なくとも一つのラベルが消去された場合のみ，頂点 i を将来の拘束伝播の起点頂点として登録するという方法である．逆に，ラベル集合 L_i が変化しなかった場合は，辺 (j,i) の辺整合性は保存されているので，頂点 i を以後の拘束伝播の起点頂点として登録する必要はない．

アルゴリズムBは，各頂点 $i(1\leq i\leq n)$ のラベル集合

```
1  repeat changed: =false;
2    for i: =1 to n do
3      for j: =1 to n do
4        if (i,j)∈E and i≠j then
5          for each x∈Li do
6            if there is no y∈Lj s.t. Rij(x,y) then
7              begin
8                delete x from Li;
9                changed: =true
10             end
11 until not changed.
```

図2 アルゴリズムA
Fig. 2 Algorithm A.

L_i および各2項関係 R_{ij} がビット列 (bit vector)[7] で表現される場合に有効性を発揮する．すなわち，集合をビット列で表すことにより，共通集合や和集合などの演算をビットごとの AND, OR で高速に処理することが可能となるからである．しかし，同型部分グラフの探索問題などにみられるように，一般にはラベル集合は任意に大きくなりうる．また，先に述べたように，拘束伝播操作は，ある辺 (i,j) についてどちらの頂点を起点とするかによって処理結果が異なる．すなわち方向性のある操作であるので，関係 R_{ij} を表すビット列を用いてこれを実現するには，辺 (i,j) について，i の各ラベルごとに一つのビット列および j の各ラベルごとに一つのビット列を用意しておく必要がある．このため，結果的に，関係 R_{ij} を各ラベルごとに再配置したビット列として二重に保持しておくことになる．

アルゴリズムBでは，拘束伝播の結果，ラベル集合が減少した頂点だけを将来の拘束伝播の起点頂点として登録するので，拘束伝播操作の総数は，アルゴリズムAに比べて大幅に減少する効果が期待できる．いま**図4**において，頂点 i,j のラベル集合をそれぞれ L_i, L_j とし，辺 (i,j) は辺整合であるとしよう．ここで，拘束伝播 prop(k,i) の結果，節 i のラベル集合が新たに L_i' になったとする．もし，L_i' が不変すなわち $L_i'=L_i$ ならば，prop(i,j) を行う必要はない．しかし，$L_i' \subsetneq L_i$ の場合は，辺 (i,j) の辺整合性が失われた可能性があるので prop(i,j) を行う必要がある．prop(i,j) の結果得られる頂点 j のラベル集合を L_j' とする．このとき，たとえ $L_j' \subsetneq L_j$ であっても，prop(j,i) は不要である．なぜなら，L_i' 内のラベルはどれも，頂点 i の適当なラベル ($\in L_i'$) と組みにして整合状態にできるという理由で削除されなかったからである．すなわち，辺整合である辺の一方の頂点のラベル集合に変化があった場合，その頂点を起点とする拘束伝播を1回行えば隣接頂点との辺整合性は再び回復される．しかしアルゴリズムBでは，ラベル集合に変化のあった場合，その頂点の情報をQに追加するようになっている．このため，上に述べた不要な拘束伝播 (図4における prop(j,i) や prop(i,k)) を排除することができない．Mackworth[6] はすでにこの性質を利用する方法を提案しているが必ずしも明確な条件として与えていない．上記のような不要な拘束伝播を避けるためには，拘束伝播が必要となる場所の指定は頂点によるのではなく，方向つきの辺を用いることが要請される (観察1)．

```
1  Q: ={j|1≦j≦n};
2  while Q≠φ do
3    begin
4      select and delete any node j from Q;
5      for i: =1 to n do
6        if (i,j)∈E and i≠j then
7          begin
8            L_i': =φ;
9            for each y∈L_j do L_i': =L_i' ∪ {x|R_ij(x,y)};
10           if L_i≠L_i∩L_i' then
11             begin
12               L_i: =L_i∩L_i';
13               Q: =Q∪{i}
14             end
15         end
16   end.
```

図3 アルゴリズムB
Fig. 3 Algorithm B.

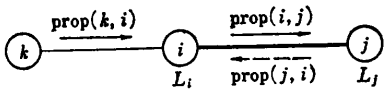

図4 拘束伝播 prop (k,i) の影響
Fig. 4 The effect of a constraint propagation prop (k,i).

つぎに，アルゴリズムBの第9行目で，$y(\in L_i)$ に対して $\{x|R_{ij}(x,y)\}$ を求めているが，もしこれが空集合ならその時点でただちに y を L_j から削除してもよいことは明らかである．この性質を用いると，prop(j,i) の処理の中で，逆向きの拘束伝播 prop(i,j) も同時に済ませることができる．すなわち，辺 (i,j) に関する互いに逆向きの拘束伝播を，一度の拘束伝播手続きで実現できる (観察2)．

3. ラベル重複度を用いたアルゴリズム

ここでは，前章で得られた二つの観察結果をとり入れ，さらに，拘束伝播によって削除される可能性のあるラベルのみを選択的に調べられるようにラベル重複度を導入した方法を新たに提案する．いま，**図5** (a) のような拘束グラフの一部があったとしよう．頂点 i,j 間には図5(b)のような2項関係 R_{ij} があるものとする．2項関係は2次元配列で表現できる．すなわち，ラベル組 (x,y) について $(x,y)\in R_{ij}$ のとき，図5(b)の関係行列の (x,y) 要素が1となっている．以後，R_{ij} を2項関係，述語および関係行列のいずれの意味にも使う．図中，$M^1_{(i,j)}, M^1_{(i,j)}$ は辺 (i,j) に

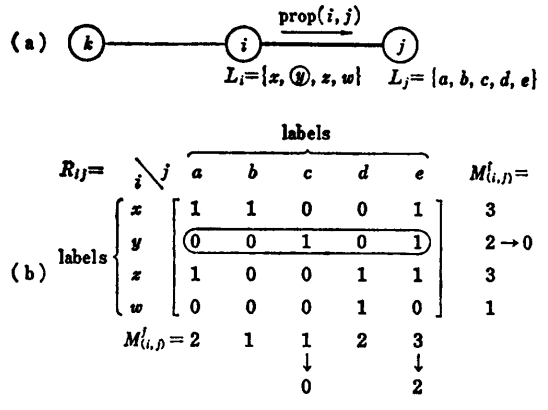

図 5 関係行列 R_{ij} とラベル多重度 $M^l_{(i,j)}, M^l_{(i,j)}$
Fig. 5 Relation matrix R_{ij} and label counts $M^l_{(i,j)}, M^l_{(i,j)}$.

関する頂点 i および j の 'ラベル重複度' といい，辺 (i, j) の関係 R_{ij} において，各ラベルが何回くり返し現れるかを示している．たとえば，$M^l_{(i,j)}(x)=3$ は，頂点 i すなわちユニット i のラベルとしてラベル x が3回現れることを表す．

ここで，いま頂点 k から i への拘束伝播 prop (k, i) の結果，i のラベル集合 L_i のうちラベル y が削除されたとしよう．これにより辺 (i, j) の辺整合性が失われた可能性があるので，拘束伝播 prop (i, j) を行う必要が生じる．このとき，ラベル重複度 $M^l_{(i,j)}$ を用いると，prop (i, j) はつぎのように容易に実現できる．まず関係行列 R_{ij} の第2行中の1の要素について，ラベル重複度 $M^l_{(i,j)}$ の対応する要素を1減ずる．図5(b)の例では，$M^l_{(i,j)}(c)$ と $M^l_{(i,j)}(e)$ が -1 される．その結果，ラベル重複度が0となったラベルはユニット j のラベルとはなりえないので，L_j から削除する．図の例では，ラベル c が L_j からとり除かれる．一般に頂点 i から j への拘束伝播 prop (i, j) では，L_i のうち新たに削除されたラベルのみに注目してラベル重複度 $M^l_{(i,j)}$ の対応する要素を0にし，さらに上の処理によって選択的にラベル重複度 $M^l_{(i,j)}$ を更新する．その結果 $M^l_{(i,j)}$ の要素で新たに0になったものがあるかどうかを調べる．もしそのような要素がないならば辺整合性は維持されており，したがって，頂点 j を起点とする拘束伝播の必要性は生じない．従来の方法では，生き残ったラベル集合 L_i と L_j とのすべてのラベルの組 $(x, y)\in L_i\times L_j$ について $R_{ij}(x, y)$ が真かどうかを調べていた．しかし，ラベル重複度を用いると，L_i から削除されたラベルの個数を l とすると，$l\cdot|L_j|$ 回のチェックで L_j からとり除くべきラベルをただちに知ることができる ($|L_j|$ は L_j の要素の数)．

```
1  for each edge e∈E do
2    mutual-prop (e); (* M*_v, M*_u and Q are set *)
3  while Q≠φ do
4    begin
5      select and delete any edge ⟨u, v⟩ from Q;
6      if prop (u, v) then
7        Q: =Q∪{⟨v, w⟩∈E|w≠u}
8    end.
```

図 6 (a) アルゴリズムCの主手続き
Fig. 6 (a) Main procedure of algorithm C.

```
1  procedure mutual-prop (e);
2    L'_u: =φ; L'_v: =φ; FLAG (e):=ON; (* e=(u, v) *)
3    for each (x, y)∈R_uv s.t. x∈L_u∧y∈L_v do
4      begin
5        add x to L'_u; M*_u(x): =M*_u(x)+1;
6        add y to L'_v; M*_v(y): =M*_v(y)+1
7      end;
8    if L'_u≠L_u then
9      begin
10       Q: =Q∪{⟨u, w⟩∈E|w≠v∧FLAG((u, w))=ON};
11       L_u: =L'_u
12     end;
13   if L'_v≠L_v then
14     begin
15       Q: =Q∪{⟨v, w⟩∈E|w≠u∧FLAG((v, w))=ON};
16       L_v: =L'_v
17     end.
```

図 6 (b) 相互拘束伝播手続き
Fig. 6 (b) Mutual propagation procedure.

```
1  procedure prop (u, v): Boolean;
2    changed: =false;
3    for each x s.t. M*_u(x)≠0∧x∈L_u do
4    begin
5      M*_u(x): =0;
6      for each y s.t. y∈L_v∧R_uv(x, y)=true do
7        begin
8          M*_v(y): =M*_v(y)−1;
9          if M*_v(y)=0 then
10           begin
11             L_v:=L_v−{y};
12             changed: =true
13           end
14       end
15   end;
16   return changed.
```

図 6 (c) 拘束伝播手続き
Fig. 6 (c) Propagation procedure.

このアルゴリズム (アルゴリズムC) を図6(a)～(c)に示す．主手続きは図6(a)である．各辺 e にはラベル重複度を保持するための1次元構造データ M^*_u, M^*_v (ただし，$e=(u, v)$) があり，主手続きに先立

って0に初期化されているものとする．各頂点 u について L_u は可能なラベルの集合を保持しており，プログラムの進行について更新されてゆく．最初は，各 L_u にはすべての候補ラベル（文献[6]では頂点整合性と呼んでいる）が格納されている．Qは，拘束伝播を行う必要のある辺と伝播の方向を頂点の順序対の形で保持する．Qは初めは空である．まず，主手続きの前半（図6(a)の第1～2行）では，拘束グラフのすべての辺について，互いに逆向きの拘束伝播を同時に行う．これを相互拘束伝播と呼び，その手続きを図6(b)の mutual-prop に示す．図6(b)では，ラベル重複度も同時に設定する．なお各辺には相互拘束伝播が済んだかどうかを示すフラグ FLAG がついており，初期設定値は OFF である．ラベル削除が起こった頂点については，その頂点を起点とした拘束伝播を行う必要があるので，その情報は頂点順序対の形でQに追加する．ただし，図6(b)の第10および15行にみられるように，すでに処理済みの辺（FLAG が ON のもの）についてのみ登録すればよい．FLAG が OFF の辺すなわち未処理の辺については，主手続き前半の for 文中において後刻，相互拘束伝播操作が施されるので，Qに登録しなくてもよい．Qに登録された各拘束伝播の項目は，主手続き後半に引き継がれる．

主手続きの後半（図6(a)の第3行以降）は，Qから順序対$\langle u, v\rangle$を次々とり出し，uからvへの拘束伝播 prop(u, v) を行う．prop(u, v) の結果，vのラベル集合 L_v に変化があれば，vから出ている辺について（ただし頂点uへ至る辺は除く）拘束伝播を行う必要が生じるので，それらをQに追加する．すでに述べたように拘束伝播は方向性のある処理なので，Q内のデータは頂点の順序対である．拘束伝播手続き prop(u, v) を図6(c)に示す．ラベル重複度を用いて R_{uv} の検査回数を少なくしている（第8～9行）．第3行は，頂点uのラベル集合 L_u のうち削除されたラベルxを取り出す部分であるが，その判定は $M_u^1(x) \neq 0 \wedge x \notin L_u$ なる条件によっている．これは，辺$e(=(u, v))$に関する頂点uのラベル重複度0がでないにもかかわらずラベル集合 L_u からは削除されているようなラベルを取り出すための条件である．この部分をさらに改良した方法としては，削除されたラベルを拘束伝播の必要な有向辺ごとに陽に記憶しておく方法がある．そうすれば第3行は，削除ラベルの集合から要素yを一つずつ取り出す操作となる．ただしこの改良を行うには，第10～13行のブロックにおいて，ラベルyを頂点vに関する削除ラベルとして覚えておき，主手続きに戻ったところ（図6(a)，第7行）で，有向辺$\langle v, w\rangle$をQに追加するとともに削除ラベルも登録するように変更しておく．

以上をまとめると，主手続き前半部では，観察2（2.2節）で述べたように，各辺ごとに，両方向の拘束伝播を mutual-prop によりまとめて行っている．また後半部では，観察1（2.2節）の性質をとり入れて，拘束伝播の必要な場所のみを頂点順序対の形で登録しておき，prop により処理する．

[例]

図7に，R_{12}, R_{13} なる拘束条件をもつ拘束グラフに主手続き前半を施した結果を示す．最初，L_1, L_2, L_3 はともに $\{a, b, c, d, e\}$ であったとし，相互拘束伝播の適用順序は辺 e_1，辺 e_2 の順と仮定する．関係行列の行または列のうち要素が全部0であるものについては，そのラベルを削除する．R_{13} の第3行（破線で囲んだ行）は，辺 e_1 の相互拘束伝播の結果，頂点1の第3ラベル（$=c$）が削除されているため，ラベル重複度 $M^1_{e_2}, M^3_{e_2}$ には寄与しない．

主手続き後半部は，前半部で得られたQの要素を処理する．本例の要素 $\langle 1, 2\rangle$ は，辺 e_2 の相互拘束伝播の結果，頂点1のラベル e が削除されたために登録されたものである．後半部の処理結果が図8である．頂点2のラベル集合からラベル d が削除され，既弛緩

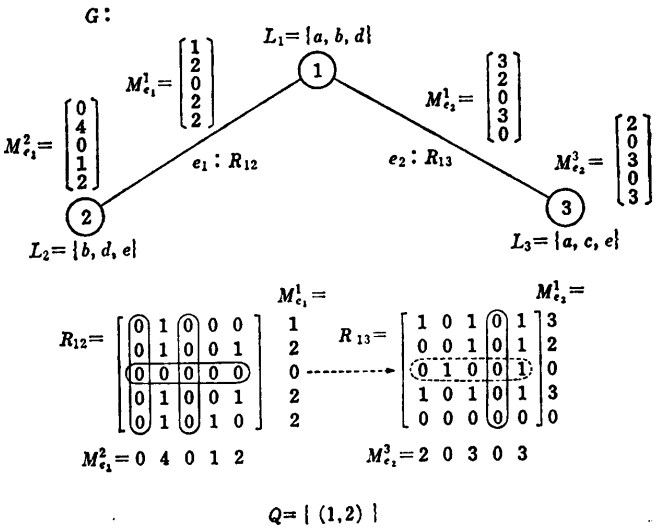

図7　相互拘束伝播（アルゴリズム前半）の実行例
Fig. 7　An example of mutual propagation.

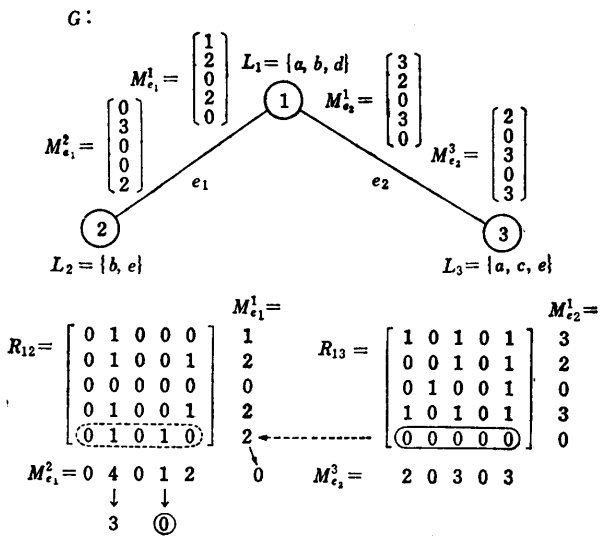

図 8 図7に prop を適用して得られる既弛緩グラフ
Fig. 8 The relaxed constraint-graph of Fig. 7 obtained after prop-procedure.

グラフが得られた．同時に，ラベル重複度 $M^1_{e_1}, M^2_{e_1}$ も更新されている．

4. 本アルゴリズムのインプリメント

ここで，本アルゴリズムをプログラム化するときに留意すべき点について述べる．また各頂点のラベル集合 L の実現には，リンク構造によるリスト，表およびビット列[7]を用いる方法がある．グラフからある部分グラフを探したり，線画からあるパターンを探す問題のように，最初の候補ラベルの個数が大きく，一方，R が疎な行列となるときは，リンク・リストが適している．本稿で述べた方法はこのような問題に向いていると考えられる．またその場合，拘束グラフは探すべき部分グラフやパターンのプロトタイプを表しており比較的小さい構造となるので，通常の隣接行列を用いることができる．しかし，ユニットの個数が多いときは，各頂点ごとに隣接頂点リストを保持するほうが，図6(a)第7行，図6(b)第10および15行の処理が速くなる．

図6(b)の第11および13行における集合の同等性の検査には，集合の大きさを表すカウンタ変数を用いる方法やビット列を用いる方法[7]がある．本アルゴリズムでは，$L'_u \subseteq L_u$ が保証されているので，前者のカウンタを用いた．ただし，第5および6行において，ラベル多重度が初めて1になった場合のみ L'_u または L'_v のカウンタを 1 増やす処理を追加しておく．こうすれば，集合の同等性の判定は，カウンタ値が等しいかどうかを調べる処理で済ませることができる．

5. 実　　験

ここでは，2章で述べたアルゴリズムA[6]，アルゴリズムB[7]，および3章で提案したアルゴリズムCの効率を計算機実験により比較する．具体的な CL 問題は，一様乱数を用いてプログラムにより発生させた．問題生成プログラムは，ユニット(拘束グラフの頂点)の個数 n，2項関係（拘束グラフの辺）の個数 m および2項関係を関係行列（図5(b)参照）で表現したときの1の要素の割合，すなわち，全ラベル対のうち2項関係を満たすものの割合 r を与えると，CL 問題を一つ発生する．発生する拘束グラフは連結グラフとなるようにしたので，$m \geq n-1$ の条件をつけた．ただし，r は m 個の 2 項関係全体に関する平均値の意味である．これは，現実の CL 問題においても，個々の2項関係の整合ラベル組の割合は互いに等しいとは限らないのが普通であるので，自然な前提と考えられる．また，三つのパラメータ n, m, r を一定にしたまま乱数の初期値を変化させることにより，多くの同種の問題を作り出すことができる．

実験では，(n, m, r) について，**表1**に示す8個の組合せの場合を調べた．それぞれの場合について 10 回ずつ CL 問題を生成した．各問題を上記三つのアルゴリズムで処理し，拘束伝播回数，関係検査回数，実行時間を調べた．頂点の個数はつねに 11 とし，各頂点の初期のラベル集合の大きさは 10 とした．したがって，最初の可能なラベルの総数は 110 であるが，拘束伝播をくり返してゆくと最終的に収束し，既弛緩グラフが得られる．**表2**は，表1の各場合について生成したそれぞれ 10 個の CL 問題に，アルゴリズムA，B，Cを適用した観測結果の，次に述べる各項目の平均値

表 1 生成した CL 問題の種類
Table 1 The cases tested in the experiments.

Case	n	m	r
1	11	10	0.15
2	11	10	0.20
3	11	15	0.20
4	11	15	0.25
5	11	15	0.30
6	11	20	0.20
7	11	20	0.25
8	11	20	0.30

表2 実験結果
Table 2 Observed values of experiments.

Case	#labels	*KP*			*KC*		
		Algorithm A	Algorithm B	Algorithm C	Algorithm A	Algorithm B	Algorithm C
1	52	84	47	26	1,635	2,135	947
2	80	70	36	19	1,909	2,503	1,002
3	0	160	117	34	1,385	2,895	1,305
4	63	184	108	74	3,566	5,769	2,222
5	91	148	77	47	3,462	5,975	2,073
6	16	192	116	64	1,918	3,348	1,522
7	84	108	60	38	2,687	4,152	1,582
8	97	84	48	28	2,226	4,045	1,546

である．測定した項目は，既弛緩時の残存ラベル総数(表では #labels)，最終状態になるまでに要した拘束伝播回数 *KP*，および関係への参照回数すなわちラベル対の整合性チェックの回数 *KC* である．本稿で提案したアルゴリズムCは，拘束伝播回数，ラベル対の整合性チェック回数のいずれについても，速い収束を実現している．アルゴリズムBの観測値 *KC* はアルゴリズムAよりむしろ5～8割大きくなっているが，これは図3の第9行の $\{x|R_{ij}(x,y)\}$ の処理において L_i の要素の全数チェックを行うことに起因している．しかし文献[7]では，この部分をビット列を用いて高速に処理することを前提としている．本来アルゴリズムBでは，拘束伝播回数 *KP* を小さくすることが目標である．本実験では行わなかったが，アルゴリズムCにおいてもビット列を用いるとラベル対の整合性チェックに続く処理は文献[7]と同様に短縮される（図6(b)の第3～7行，図6(c)の第6および11行）．また，各CL問題の処理に要した総演算時間（ユーザCPU時間）は，アルゴリズムBはAの1.5～2倍かかった．これはビット列を用いなかったためである．本実験のようにラベル個数の少ない場合はビット列を用いるべきであろう．アルゴリズムCはBと同様にビット列を用いていないが，アルゴリズムAの約6割の時間で処理できた．ビット列を用いるとさらに短縮できると思われるが，本方法はラベル総数が大となる問題に適用することも考えているので本実験では用いなかった．なお，総処理時間については，ラベル対の整合性チェックの処理時間の影響をうけるので，本実験のみから結論づけることはできない．本実験では，2項関係を関係行列で表したので，ラベル対の整合性チェックは配列要素へのアクセス処理となる．しかし，ビット列などを用いた高速処理を採用したときと，ラベル集合が大きいため関係をマルチ・リストで表現したときとでは，アルゴリズム間の処理時間比は大きく異なると思われる．すなわち，前者の場合は表2の *KP* 値が，また後者の場合は表2の *KC* 値が，より近い処理時間比を与えることになろう．なお実験は，FORTRAN 77相当を用いて Perkin-Elmer 3220上で行った．

6. むすび

CL問題は，問題の対象の構成部分である要素（ユニット）とその解釈の候補（ラベル）および要素間に成り立つべき解釈の拘束条件によって与えられる．拘束条件は，一般に要素の多項関係の集合となるが[3]，等価な2項関係の集合に変換することができる[4),5]．本稿では，2項関係のみに限り，CL問題の拘束グラフ表現を中心に考察した．二つのユニット間にラベルの2項関係が存在する場合，おのおののユニットのラベル集合をこの2項関係をもとに絞ってゆくアルゴリズム，いわゆる辺整合アルゴリズムについて，従来の方法を考察した．また，2項関係においてラベルの出現する回数，すなわちラベル重複度を導入したアルゴリズムを提案し，計算機実験によりその有効性を評価した．

辺整合アルゴリズムは，CL問題の最終的な解を与えるものではないが，深さ優先探索などの探索技法において無駄なバックトラッキングのくり返し現象(thrashing)を防止するのに有効である．また，本稿で提案したラベル重複度は，Freuder[12]のように拘束伝播を拡張して最終解を得るようにした方法にも適用できる．CL問題は，ユニットの個数に比べて，①関係の個数や次元が大で，かつラベル集合も大の場合と，②関係の個数や次元が小さく（たとえば2項関係

のみ)，かつラベル集合も小の場合の二つの方向に分類できる．同型な部分グラフの探索問題は前者の例であり，グラフの4色ぬり分けやNクイーン問題などは後者の例である．本稿で考察した拘束伝播の方法は可能なラベルの集合を絞ってゆく手続きであり，主として前者の性質をもつCL問題に適していると思われる．しかし，CL問題として把握できる問題は多く知られており(文献13)など)，CL問題の定義に必要なユニット，ラベルおよび拘束条件の三つについて統一的な観点から整理し，多様なCL問題の性質を明らかにすることは今後の課題である．

参考文献

1) Waltz, D. L.: Understanding Line Drawings of Scenes with Shadows, in P. H. Winston (ed.): *The Psychology of Computer Vision*, McGraw-Hill, New York (1975) (白井，杉原(訳)：コンピュータービジョンの心理，産業図書，東京(1979))．
2) Ullmann, J. R.: An Algorithm for Subgraph Isomorphism, *J. ACM*, Vol. 23, No. 1, pp. 31-42 (1976).
3) Haralick, R. M. and Shapiro, L. G.: The Consistent Labeling Problem, Part I, *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. PAMI-1, No. 2, pp. 173-184 (1979); Part II, *ibid.*, Vol. PAMI-2, No. 3, pp. 193-203 (1980).
4) Nudel, B.: Consistent-Labeling Problems and Their Algorithms: Expected-Complexities and Theory-Based Heuristics, *Artif. Intell.*, Vol. 21, No. 1 & 2, pp. 135-178 (1983).
5) Montanari, U.: Networks of Constraints: Fundamental Properties and Applications to Picture Processing, *Inf. Sci.*, Vol. 7, No. 2, pp. 95-132 (1974).
6) Mackworth, A. K.: Consistency in Networks of Relations, *Artif. Intell.*, Vol. 8, No. 1, pp. 99-118 (1977).
7) McGregor, J. J.: Relational Consistency Algorithms and Their Application in Finding Subgraph and Graph Isomorphisms, *Inf. Sci.*, Vol. 19, No. 3, pp. 229-250 (1979).
8) Rosenfeld, A., Hummel, R. A. and Zucker, S. W.: Scene Labeling by Relaxation Operations, *IEEE Trans. Syst. Man, Cybern.*, Vol. SMC-6, No. 6, pp. 420-433 (1976).
9) 白井，辻井：人工知能，相磯他編，岩波講座情報科学 22，岩波書店，東京 (1982)．
10) Gaschnig, J.: A General Backtrack Algorithm That Eliminates Most Redundant Tests, Proc. Int. Conf. on Artif. Intell., Cambridge, Mass., pp. 457 (1977).
11) Haralick, R. M. and Elliott, G. L.: Increasing Tree Search Efficiency for Constraint Satisfaction Problems, *Artif. Intell.*, Vol. 14, No. 3, pp. 263-313 (1980).
12) Freuder, E. C.: Synthesizing Constraint Expressions, *Comm. ACM*, Vol. 21, No. 11, pp. 958-966 (1978).
13) Ballard, D. H. and Brown, C. M.: *Computer Vision*, SS. 9. 5, 12. 4, Prentice-Hall, Englewood Cliffs (1982).

(昭和58年9月21日受付)
(昭和59年6月19日採録)