CSPにおけるノードの値の重みとアーク重みの相互変換

1M－4

斎藤 逸郎　蔡 東風　石塚 満
東京大学工学部電子情報工学科
e-mail: saito@miv.t.u-tokyo.ac.jp

1　はじめに

重みつきCSPの重みには，ノードの値に重みがついたものとアークに重みがついたものが存在する[2]．両者の重みを相互に変換する事のより，どちらか一方の重みだけを対象とした解法を用いる事ができ，ひいては問題の簡易化や問題解決の効率化が可能となる．この重みの相互変換の手法について提唱する．

2　重みの変換

変換方法に触れる前に変数について定義をしておく．

各々のノード変数 $x_1,\ldots,x_n$ について変域を $D_1,\ldots,D_n$ とした時ノードおよびアークに対する制約をそれぞれ，

ノードに対する制約 $\{u \in D_i | p_i(u)\}$

アークに対する制約 $\{(u,v) \in D_i \times D_j | p_{i-j}(u,v)\}$

と記す．

2.1　ノードの値の重みからアーク重みへの変換

i 番目のノードを $Node_i$ とする．$Node_i$ の要素の数を m_i とした時，$k(0 \leq k \leq m_i)$ 番目の要素を $u_{ik}, D_i = \{u_{ik}\}$ とする．$Node_i$ から $Node_j$ へ出ているアークに対する制約を $\{(u_{ik},v_{jl}) \in D_i \times D_j | p_{i-j}(u_{ik},v_{jl})\}, D_i = \{u_{ik}\}, D_j = \{v_{jl}\}$ とし，$Node_i$ から出ているアークのうち，制約中に u_{ik} を含むアークの数を $n(u_{ik})$ とする．制約ネットワーク中の $n(u_{ik})$ の最大値を n_{max} とする．また u_{ik} の重みを $W(u_{ik})$ とする．

この場合以下の手順によりノードの値の重みからアーク重みへ変換可能である．

1. アークに対する制約 $\{(u_{ik},v_{jl}) \in D_i \times D_j | p_{i-j}(u_{ik},v_{jl})\}, D_i = \{u_{ik}\}, D_j = \{v_{jl}\}$ について，矛盾を含まない形に書換える．

   図1では a,b,c,d,e,f,g が各ノードの要素であり，(a,c) の様に書かれているのが制約である．

2. 要素が一つのノード $Node_\alpha$ を定め，その要素を α とし，重みは0とする．

3. $Node_a$ より $Node_\alpha$ に対して $\{(u_{ak},\alpha) \in D_a \times \{\alpha\} | p_{a-A}(u_{ak},\alpha)\}$ なる制約を持つアークをはる．この場合制約候補の数は m_a 個になる．

   図2で α を要素として持つノードから a,b を要素としてもつノード，e,f を要素としてもつノード，g を要素としてもつノードに出ているアークが，これである．

4. 制約 $(u_{ak},v_{jl}) \in D_a \times D_j | p_{a-j}(u_{ak},v_{jl})$ の重みに

$$\frac{W(u_{ak}) + W(v_{jl})}{n_{max}}$$

   を加え，制約 $(u_{ak},\alpha) \in D_a \times \{\alpha\} | p_{a-A}(u_{ak},\alpha)$ の重みに

$$\frac{(n_{max} - n(u_{ak}))W(u_{ak})}{n_{max}}$$

   を加える．

   図3で '－＞' で示されているのがアーク重みである．ノードの要素 a,b について見てみると，制約中に a を含むアークの数は2（(a,c) を含むアークと (a,e) を含むアーク）であり，制約中に b を含むアークの数は1（(b,c) を含むアーク）である．また各要素について制約中に要素が含まれるアークの数を見てみると，制約中に c もしくは d を含むアークの数はともに3（(a,c) を含むアーク，(c,e) を含むアーク，(c,d) を含むアークと (c,d) を含むアーク，(d,e) を含むアーク，(d,g) を含むアーク）であり，これが最大であり，$n_{max} = 3$ となる．

   これらより各アークの重みは図3の様になる．

5. ノードの重みをすべて0とする．

Conversion of node value weight and arc constraint weight on CSP

Itsuro SAITO, Dongfeng CAI, Mitsuru ISHIZUKA
School of Info. & Commun. Eng., Univ. of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo, 113, JAPAN

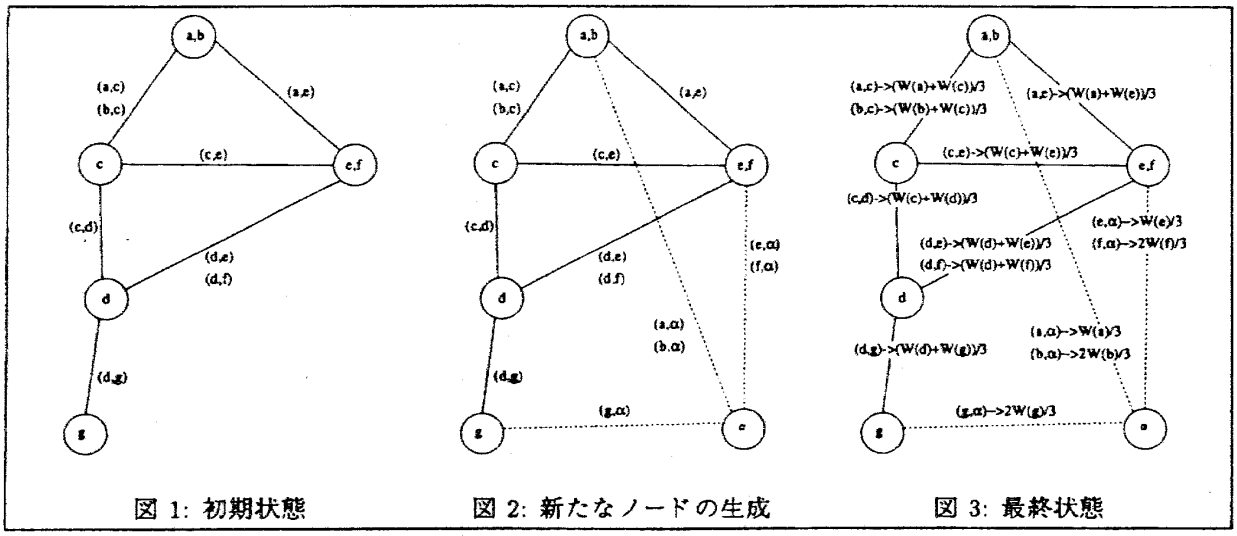

図 1: 初期状態　図 2: 新たなノードの生成　図 3: 最終状態

図 4: ノードの値の重みからアーク重みへの変換

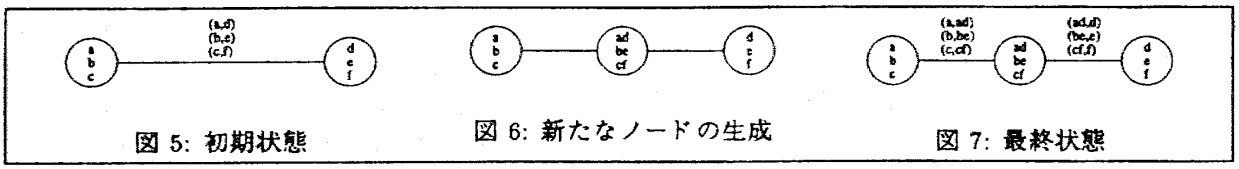

図 5: 初期状態　図 6: 新たなノードの生成　図 7: 最終状態

図 8: アーク重みからノードの値の重みへの変換

2.2 アーク重みからノードの値の重みへの変換

i 番目のノードを $Node_i$ とする．$Node_i$ から $Node_j$ へ出ているアークに対する制約を $\{(u_{ik}, v_{jl}) \in D_i \times D_j | p_{i-j}(u_{ik}, v_{jl})\}, D_i = \{u_{ik}\}, D_j = \{v_{jl}\}$ とする．

この場合以下の手順によりアーク重みからノードの値の重みへ変換可能である．

1. アークに対する制約 $\{(u_{ik}, v_{jl}) \in D_i \times D_j | p_{i-j}(u_{ik}, v_{jl})\}, D_i = \{u_{ik}\}, D_j = \{v_{jl}\}$ について，矛盾を含まない形に書換える．

   図 5 では a, b, c, d, e, f が各ノードの要素であり，(a, d) の様に書かれているのが制約である．

2. $Node_i$ から $Node_j$ のアークの間に $Node_{ij}$ なるノードを定め，その要素を $p_{i-j}(u_{ik}, v_{jl})|w_{ikjl}$ とする．つまり $Node_{ij}$ の要素の数はアークに対する制約 $p_{i-j}(u_{ik}, v_{jl})$ の要素の数と等しくなるようにする．また w_{ikjl} の重み $W(w_{ikjl})$ は，アークに対する制約 $\{(u_{ik}, v_{jl}) | p_{i-j}(u_{ik}, v_{jl})\}$ の重みと等しくする．

   図 6 では ad, be, cf を要素としてもつノードが新たに生成されたノードである．それぞれの重みは，$W(ad)$ は $p(a, d)$ の重みと，$W(be)$ は $p(b, e)$ の重みと，$W(cf)$ は $p(c, f)$ の重みと等しくなるようにする．

3. $Node_i$ から $Node_{ij}$ へのアークに対して，$\{(u_{ik}, w_{ikji}) | p_{i-ij}(u_{ik}, w_{ikjl})\}$ なる制約を与え，$Node_{ij}$ から $Node_j$ へのアークに対して，$\{(w_{ikjl}, v_{jl}) | p_{ij-j}(w_{ikjl}, v_{jl})\}$ なる制約を与える．

4. アークの重みをすべて 0 とする．

3 まとめ

以上によりアーク重みとノードの値の重みとの相互変換ができるようになる．このやり方では，特にアーク重みからノードの値への重みの変換で，ノード数が大幅に増えるという問題が存在する．この問題を解決する事が今後の課題となり得る．

参考文献

[1] 石塚満: 知識の表現と高速推論, 丸善, 1996.

[2] 蔡 東風, 石塚満: 局所探索に基づく重み付き制約充足問題の効率的解法, 第 54 回情報処理学会 全大, No.2, pp.327–328, 1997.

[3] Hoong Chuin LAU: A New Approach for Weighted Constraint Satisfaction: Theoretical and Computational Results, *Constraint Satisfaction, CP'96*, pp.323–337, 1996.