6F−7

リートネットワークによる併合法の実現*

牛田修司　　佐塚秀人†
NTTデータ通信(株)　久留米工業大学‡

1　はじめに

制約問題解決は、画像処理、計画/設計問題など適用範囲がひろく、最近その研究が盛んである。併合法は制約充足方法の1つであり、本稿では併合法を従来OPS5系のシェルで用いられてきたRete Match Algorithm[1]を用いて実現する方法について述べる。

2　併合法とリートネットワーク

併合法での制約ネットワークとは、解の候補の集合を持っているノードと、それらのノード間を制約で結んだネットワークである(図1-a)(以下併合ネットワーク)。併合法における制約充足とは図のような併合ネットワーク上で隣接するノードを併合していき、最終的に1つのノードにすることである(図1)。

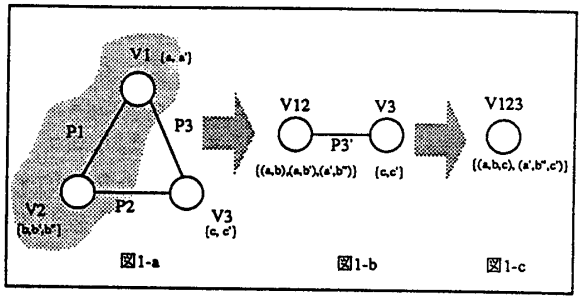

図1: 併合ネットワークと制約充足

一方リートネットワークは、各ノードが条件を持っており、トークンと呼ばれる変数がその条件を満たす時に次のノードへ透過させる。ノードにはイントラノードとインターノードがあり、イントラノードは1変数に関する条件を持つ。インターノードは、2変数にまたがる条件を記述してあり、2つのノードからの入力を持つ。インターノードでは片方から入力されたある変数を評価する時は他方から入力されたもの全てに対して評価を行なう必要があるため、入力された変数を保存しておくメモリを持っている。また、最終的に条件を全て満たす変数の組はターミナルノードに保存される。リートでは変数の値を書き換えることによってネットワークの状態を変化させ、推論を進める。リートは次のようにして全体の計算量を抑えている。

1. 同じ条件はノードを共有する
2. 中間結果を各ノードに保存することにより、変数の値が変化した時は、その変化分のみを計算する

3　リートネットワークへの展開

併合法におけるノードの併合という過程は、ある制約を満たす変数の組をみつけることであり、これはリートネットワークのインターノードと同じである。リートのインターノードは$P(x,y)$を満たす(x,y)をすべてみつけることであり、インターノードで結合される2つの論理式はAnd結合である[1]。

従って、図1-aの制約が全てAnd結合で記述されている場合、各制約はインターノードに展開できる。併合ネットワークのノード上の変数はリートではトークンとして扱えるので併合ネットワークはリートネットワークに展開できる。図2は図1-aをリートに展開したものである。

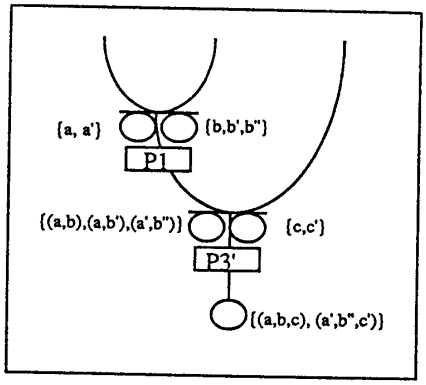

図2: リートネットワークへの展開

また、$\forall x, P(x,y)$の論理式をリートで展開する場合、$P(x,y)$を満たすインターノードの下にオールノードを設ければよい。オールノードは1つのメモリを持ち、上のインターノードの結果を保存する。オールノードの処理は、新しいトークン(x_i,y_j)が入ってきた場合、メモリ内からx_iを含めた集合$\{\cup x_k; P(x_k,y_j)\}$を求め、これがxの値域と等しい場合、トークン(x,y_j)を下のノードに流す(図3)。また、xの値域が変化した場合、全てのy_jに

[1] xが$P_1(x_1,x_2)$を、yが$P_2(y_1,y_2)$を満たしている場合、インターノード$P(x,y)$を満たすとは、$P_1(x_1,x_2) \wedge P_2(y_1,y_2) \wedge P(x,y)$を満たすこと。

*Implementation of Merge Method by Rete-network
†Shuji USHIDA, Hideto SAZUKA
‡NTT DATA COMMUNICATIONS SYSTEMS CORP., Kurume Institute of Technology

対して、$\{\cup x_i; P(x_i, y_j)\}$ を求め、その集合が値域と一致しない場合は (x, y_j) のマイナストークンを流して削除する。

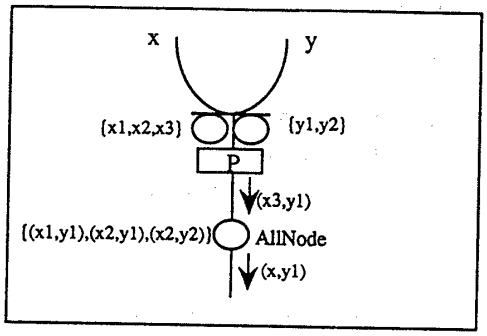

図 3: オールノード

通常リートネットワークでノットノードと呼ばれるノードはこのオールノードの変形であり、上で2つに分けて処理を行なったインターノードとオールノードはノットノード同様1つのノードで処理できる。

両者の違いは併合ネットワークが解候補集合を固定して制約を伝播するのに対し、リートでは逆に制約が固定で変数がネットワーク上を流れることであるが、併合ネットワークをリートネットワークに展開することによって、解候補の集合が動的に変化するような問題も差分だけの計算ですみ、計算量を抑えられる。

4 動的な制約の変化

制約充足問題では動的に制約の強化・緩和・削除などを行なうことによって over constraint/under constraint に対応し、問題解決の補強を行なうが、これはリートではネットワークをダイナミックに再構成・再評価を行なうことによって実現される。

リートで制約の強化・緩和・削除を扱う場合、主に次のように分類される

1. 既存の併合ネットワークが変化する場合

   (a) ノード間の制約条件が変化
   併合ネットワークの構造は変わらず、各ノード間の制約のみが変化する場合、その条件に対応するリートのノードの条件を変えるて、それ以降のノードを再評価する

   (b) ノードの数が増減
   併合ネットワークのノードの数が変化する場合

   i. ノードが減った場合
   その変数がでてくる一番上のインターノードを無くし、メモリにある相手側のトークンを下に流す。

   ii. ノードが増えた場合
   インターノードを追加する。一番単純なのは今までの ターミナルノードのところにインターノードを追加することであるが、できるだけノードを共有することや、探索空間が狭くなる場所などを考慮して追加する必要がある。

2. 新規に併合ネットワークが追加される場合

   別の併合ネットワークを追加する場合、その分のリートネットワークをダイナミックに構成し、全体に追加する。 その際、図4-aのように、既存のノード・制約の組合せ(ノード上の変数の値域は違っていても良い)と同じものがある場合は、リートに展開する時、既存のインターノードと共有するようにする(図4-b)。これは、各インターノードのメモリには計算結果が保存されているので、それを利用することによって計算量を減らすためである。

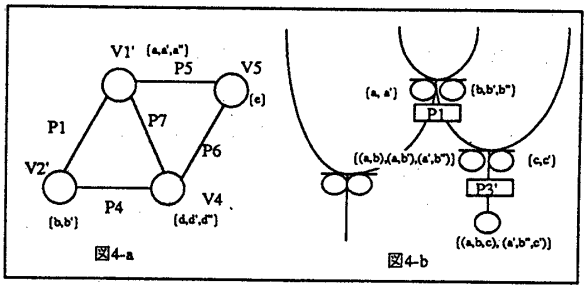

図 4: 制約の変化と探索木の再構成

また、制約の強化・緩和・削除によって競合集合が変わるので、これを用いてメタ知識を記述することによってリートの制御も行なえる。

5 おわりに

リートネットワークを用いた併合法による制約充足の方法について述べたが、これはルール型言語と同じ枠組の中で制約も扱えるように拡張できることである。今後の課題としては、変数の領域が不定 / 連続なものはを扱えるようにすることと、リートネットワークをいかに効率良く作成するかを検討することである。

参考文献

[1] Charles L.Forgy: RETE: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem, Artificial Intelligence No.19(1982), pp.17-37

[2] Raimund Seidel: A NEW METHOD FOR SOLVING CONSTRAINT SATISFACTION PROBLEMS IJCAI,7th (1981), pp.338-342

[3] 西原精一, 松尾嘉和, 池田克夫 : 既整合ラベリング問題における併合法の最適化と効率評価 人工知能学会誌,vol.3,No.2 (1988), pp196-205