3E2-06

サイクルカットセットを用いた分散制約充足アルゴリズム

A Cycle-Cutset-based Algorithm for Solving Distributed CSP

松下 俊伸 Toshinobu Matsushita　横尾 真 Makoto Yokoo　岩崎 敦 Atsushi Iwasaki

九州大学大学院 システム情報科学研究院
Graduate School of Information Science and Electrical Engineering, Kyushu University

This paper develops a new algorithm for solving distributed Constraint Satisfaction Problems (disCSPs) called cycle-cutset-based asynchronous backtracking (CC-ABT), since it is based on the traditional asynchronous backtracking (ABT) algorithm, while it utilizes a notion called cycle-cutset to identify a difficult part of the problem. A disCSP is a general framework that can formalize various application problems in multi-agent systems. A cycle-cutset is a set of variables that cut all cycles, i.e., after removing this set, the remaining constraint network becomes acyclic. Since an acyclic constraint network can be solved in a polynomial time, cycle-cutset-based methods are extensively examined for centralized CSPs. However, so far, this idea is not widely used in disCSP studies. In our newly developed CC-ABT algorithm, agents identify a cycle-cutset using a simple distributed algorithm, then, the agents in the cycle-cutset perform the asynchronous backtracking, while the agents in the remaining acyclic parts concurrently perform directed-arc-consistency.

1. はじめに

マルチエージェントシステムとは，複数の自律的に動作するエージェントの相互作用，特にこれらのエージェント間の協調に関する人工知能の研究の一分野である． 近年の計算機の小型化，低価格化，および計算機ネットワーク技術の進歩により，分散計算機環境が急速に普及しつつあり，このような自律的なエージェントに関する研究の必要性は非常に大きくなっている． このため，マルチエージェントシステムは人工知能の中でも非常に活発な分野となっている．

一方，制約充足問題 [Dechter 03] とは，有限で離散的な領域から値を取る複数の変数に対して，制約を満足する値の割当を発見する問題であり，人工知能の様々な問題がこの問題により定式化できることが知られている．著者らは制約充足問題の変数，制約が複数のエージェントに分散された問題として分散制約充足問題を定義し [横尾 92, Yokoo 01]，マルチエージェントシステムの様々な応用問題が分散制約充足問題として定式化可能であることを示した．このように，分散人工知能，マルチエージェントシステムで生じる様々な問題が分散制約充足問題として定式化可能であるため，分散制約充足問題を解くための分散アルゴリズムは，エージェント間の協調を実現するための重要なインフラストラクチャであると考えることができる．

制約充足問題においては，サイクルカットセットと呼ばれる概念を利用した様々なアルゴリズムが提案されている [Dechter 03]．一方，分散制約充足問題を解くアルゴリズムにおいては，サイクルカットセットのアイデアは広く用いられるに至っていない．本論文では，サイクルカットセットを利用した新しい分散制約充足アルゴリズムである サイクルカットセットに基づく非同期バックトラッキング (Cycle-Cutset-base Asynchronous BackTracking, CC-ABT) を提案する．本アルゴリズムの特徴は，サイクルカットセットを発見することにより，問題の難しい部分を抽出して，変数の優先順位を決定する点である．

A: 松下 俊伸，九州大学，812-8581 福岡県福岡市東区箱崎 6-10-1，(092)642-4065，matusita@lang.is.kyushu-u.ac.jp

2. 準備

2.1 制約充足問題

制約充足問題（CSP）は一般に次のように定義される．n 個の変数 $x_1, x_2, \ldots, x_n$ と，変数のそれぞれが値をとる有限で離散的な領域 $D_1, D_2, \ldots, D_n$，および制約の集合が存在する．本論文では制約は述語によって内包的に定義されるとする．すなわち，制約 $p_k(x_{k1}, \ldots, x_{kj})$ は，直積 $D_{k1} \times \ldots \times D_{kj}$ に対して定義され，これらの変数の値が互いに整合のとれている場合に真となる．制約充足問題の解を求めることは，すべての制約を満足する変数の値の組を求めることである．

制約がすべて二項関係である場合，制約充足問題は変数をノード，制約をリンクとする制約ネットワークにより表現できる．制約ネットワークから変数の部分集合 C，および C の関連する制約を取り除いた場合に，残った変数と制約がサイクル (閉路) を持たない場合，変数の部分集合 C をサイクルカットセットと呼ぶ [Dechter 03]．制約ネットワークがサイクルを持たない場合，その制約充足問題は多項式時間で解くことができる．サイクルカットセットを利用した集中型の制約充足問題を解くアルゴリズムとして，CC アルゴリズム [Dechter 03] が知られている．サイクルカットセットを C，残りの変数を A とした場合，CC アルゴリズムは以下のように動作する．

CC アルゴリズムでは，まず，C に属する変数間の制約をすべて満足するように，C に属する変数に対する値の割当てを，通常の制約充足問題の解法 (例えばバックトラッキング) を用いて求める．次に，A に属する変数に関して，C との制約，および A の間の制約をすべて満足する変数への値の割当てを求める．このような割当てが存在するかどうかは directed-arc-consistency [Dechter 03] を用いることにより，多項式時間で決定できる．そのような A に対する値の割当てが存在しない場合，C に関して他の割当てを求め，上記の処理を繰り返す．

上記アルゴリズムの複雑度は，サイクルカットセットのサイズに対して指数的であるため，より小さいサイクルカットセットを求めることが望ましいが，最小のサイクルカットセットを見つける問題は NP-完全であることが知られている．

2.2 分散制約充足問題

分散制約充足問題とは，制約充足問題の変数が複数のエージェントに分散された問題である．本論文では以下のエージェント間通信のモデルを仮定する．

- エージェント間通信はメッセージ通信によってなされる．
- エージェントは，他のエージェントのアドレスを知っている場合に限りそのエージェントにメッセージを送信できる．
- メッセージの遅延は有限であるが，遅延時間の上限は分かっていない．
- 任意の二つのエージェントの組み合わせに関して，送信されたメッセージの順序は保存される．

各エージェントは自分の持つ変数の値を決定しようとするが, 異なるエージェントの持つ変数間に制約がある．エージェントの目的は，エージェント間の制約をすべて満足する値の割当を見つけることである．

以下，アルゴリズムの説明を行う際に，簡単のため次の仮定をおく． これらの仮定を緩和し，アルゴリズムを一般の場合に拡張することは容易である．

- 各エージェントの持つ変数は唯一である．
- 各エージェントは自分に属する変数が関係する制約をすべて知っている．
- エージェント間の制約はすべて二項関係（binary）である．

すべての制約が二項関係である分散制約充足問題はネットワークを用いて表現できる．すなわち，変数がノードに対応し，ノード間のリンクが制約に対応する． また，各エージェントは唯一の変数を持つと仮定しているため，各ノードはエージェントに対応すると考えることもできる．以下，エージェント i と変数 x_i に関して共通の識別子 x_i を用いる．すべてのエージェントおよび変数にはユニークな識別子が与えられていると仮定する．各エージェントに対して，リンクで接続されたエージェントの集合をそのエージェントの近傍と呼ぶ．

[Yokoo 98a] では，非同期バックトラッキングアルゴリズム (Asynchronous BackTracking, ABT) と呼ばれる，分散制約充足問題を解く基本的なアルゴリズムが示されている．ABT では，各エージェントは非同期，並行に動作するが，アルゴリズムの完全性は保証されている．

3. サイクルカットセットに基づく非同期バックトラッキング

3.1 アルゴリズムの概要

本章では，サイクルカットセットを利用した新しい分散制約充足アルゴリズムである サイクルカットセットに基づく非同期バックトラッキング (Cycle-Cutset-base Asynchronous BackTracking, CC-ABT) について説明する．アルゴリズムは以下の手順からなる．

1. エージェントは次節に示す分散サイクルカットセット検出アルゴリズムを実行し，各エージェントは自分がサイクルカットセット C に含まれるか，その他の閉路のない部分 A に含まれるかを決定する．
2. C に含まれるエージェントは非同期バックトラッキング (ABT) アルゴリズムを実行する．
3. A に含まれるエージェントは，C に含まれる変数との制約を満足する値の組合せを，directed-arc-consistency を達成することにより発見する．そのような値の組合せが存在しない場合には，A 中のエージェントから C 中のエージェントに対して，新しい制約条件に関する情報 (nogood) が送信される．

本アルゴリズムの特徴は，サイクルカットセット中の変数の値を決定する処理と，サイクルカットセット中の他の変数の値を決定する処理が非同期，並行に実行される点である．非同期バックトラッキングアルゴリズムが完全であることから，CC-ABT が完全であることが導かれる．

次節で，CC-ABT の主要部分である分散サイクルカットセット検出アルゴリズムについて解説する．

3.2 分散サイクルカットセット検出アルゴリズム

本論文で提案する分散サイクルカットセット検出アルゴリズムは，[Jagota 97] で示されている分散アルゴリズムをベースにしているが，[Jagota 97] では，一度に一つのエージェントのみが動作することが保証される分散計算環境を仮定しており，本論文での問題設定では直接用いることができない．このため，本論文では，分散ブレイクアウトアルゴリズム [Yokoo 96, 横尾 98b] で用いられている，近傍のエージェント間の相互排除と同様の方法を用いて，近傍でないエージェントが同時に動作することを可能とすると同時に，相互排除時にヒューリスティックを導入して，より小さいサイクルカットセットを求めることを可能にしている．

本アルゴリズムの実行中に，各エージェントは C (サイクルカットセットに属する), A (C 以外の木構造の部分に属する), U (どちらに属するかまだ決定していない) の 3 種類の状態を取り，初期状態は U である．

アルゴリズムの動作は以下の通りである．

1. 近傍のエージェントと，現在の状態に関する情報を交換する．もし現在の状態が U であれば，状態が A である近傍のエージェントの個数，および すべての近傍のエージェントの個数を交換する．
2. もし現在の状態が A および C である場合：現在の状態を保持する．
3. 現在の状態が U であり，近傍に 2 つ以上の A がある場合：状態を C に変更する．
4. 現在の状態が U であり，近傍に唯一の A がある場合：
   (a) 近傍中に，状態が U で，かつ近傍に唯一の A があるエージェントが他に存在しない場合：状態を A に変更する．
   (b) 近傍中に，状態が U で，かつ近傍に唯一の A があるエージェントが存在し，かつこれらのエージェント中で，近傍のエージェントの個数が最小である場合 (タイブレークはエージェントの識別子の辞書式順序を用いる)：状態を A に変更する．
   (c) その他の場合：現在の状態 U を保持する．
5. 現在の状態が U であり，近傍に A が存在しない場合：

(a) 近傍中に，状態が U で近傍に 2 つ以上の A を持つエージェント, もしくは 状態が U で，かつ近傍に唯一の A があるエージェントが存在せず，かつ，近傍中の状態が U であるエージェント中で，近傍のエージェントの数が最小である場合 (タイブレークはエージェントの識別子の辞書式順序を用いる)：状態を A に変更する．

(b) その他の場合：現在の状態 U を保持する．

上記アルゴリズムで，A に属するエージェントは，A になる時点で近傍に他の A が存在しなかった場合，木構造でのルートとなり，その他の場合は近傍に唯一存在した A を親ノードとする．

サイクルカットセットのサイズを小さくするためのヒューリスティックとして，近傍の変数の個数が大きい変数を優先してサイクルカットセットに加えるという方法が知られている [Dechter 03]．本アルゴリズムでは，木構造に加えるエージェントを選択する際に，近傍のエージェントの個数が小さいものを優先して選択することにより，同様な効果を実現している．

本アルゴリズムに関して次の定理が成立する．

定理 1 本アルゴリズムが終了し，すべてのエージェントが A もしくは C の状態となった場合，A のエージェントからなる制約ネットワークは閉路を持たない．

証明： エージェントが U から A に変化するのはアルゴリズムの 4-(a)，4-(b), もしくは 5-(a) の場合のみであり，いずれの場合も，近傍での相互排除により，近傍中で同時に複数のエージェントが A に変化することはない．また，どちらの場合でも，A になるエージェントの近傍には 0 ないし唯一の A しか存在しないので，閉路が構成されることはありえない． □

定理 2 本アルゴリズムを実行した場合, いずれはすべてのエージェントが A もしくは C の状態となる定常状態に到達する．

証明： 各サイクルで，一つないし複数の状態 U のエージェントが存在する場合，これらのエージェント中で，少なくとも一つのエージェントが状態を A ないし C に変更することを示す．これにより，U の個数は単調に減少し，いずれはすべてのエージェントが A もしくは C となる定常状態に到達することが導かれる．

以下，一つないし複数の状態 U のエージェントが存在するが，いずれも状態を変更しないと仮定して矛盾を導く．この場合，すべての U のエージェントで，4-(c) もしくは 5-(b) の条件が成立していることになる．

まず，4-(c) が成立するエージェントが少なくとも一つ存在する，すなわち，状態が U で，近傍に唯一の A を持つエージェントが少なくとも一つ存在する場合を考える．この場合，このようなエージェント中で，近傍のエージェントの個数が最小，かつ辞書式順序で最初となるエージェント x を選択する．x に関して，4-(c) の条件が成立するため，近傍に唯一の A を持ち，かつ，x よりも近傍のエージェントの個数が小さいか，辞書式順序で先になるエージェントが存在することになるが，これは x の選び方と矛盾する．

次に，すべての状態が U のエージェントに関して，4-(c) の条件が成立しない，すなわち, 各エージェントで 5-(b) の条件が成立している場合を考える．このようなエージェント中で，近傍のエージェントの個数が最小となるエージェント x を選択する．x に関して，5-(b) の条件が成立し，かつ，4-(c) の条件が成立するエージェントは存在しないという仮定から，状態が U で，x より近傍のエージェントの個数が小さいか，辞書式順序で先になるエージェントが存在することになるが，これは x の選び方と矛盾する．

上記より，一つないし複数の状態 U のエージェントが存在する場合，少なくとも一つのエージェントが状態を A ないし C に変更することが導かれた． □

4. まとめ

本論文では，サイクルカットセットを利用した分散制約充足問題を解くアルゴリズムである CC-ABT アルゴリズムを提案した．本アルゴリズムの特徴は，サイクルカットセットを発見することにより，問題の難しい部分を抽出して，変数の優先順位を決定する点である．CC-ABT の主要な部分は，分散サイクルカットセット検出アルゴリズムであり，近傍のエージェント間の相互排除により，近傍でないエージェントが同時に動作することを可能とすると同時に，相互排除時にヒューリスティックを導入して，より小さいサイクルカットセットを求めることを可能にしている．

現在，CC-ABT アルゴリズムの実装を終え，様々な例題に関して評価実験を進めている．

参考文献

[Dechter 03] Dechter, R.: *Constraint Processing*, Morgan Kaufmann (2003).

[Jagota 97] Jagota, A. and Dechter, R.: Simple distributed algorithms for the cycle cutset problem, in *Proceedings of the 1997 ACM symposium on Applied computing table of contents*, pp. 366 – 373 (1997).

[横尾 92] 横尾, エドモンド H., 石田, 桑原：分散制約充足による分散協調問題解決の定式化とその解法, 電子情報通信学会論文誌, Vol. J-75 D-I, No. 8, pp. 704–713 (1992).

[Yokoo 96] Yokoo, M. and Hirayama, K.: Distributed Breakout Algorithm for Solving Distributed Constraint Satisfaction Problems, in *Proceedings of the Second International Conference on Multi-Agent Systems*, pp. 401–408, MIT Press (1996).

[Yokoo 98a] Yokoo, M., Durfee, E. H., Ishida, T., and Kuwabara, K.: The Distributed constraint satisfaction problem: formalization and algorithms, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 10, No. 5, pp. 673–685 (1998).

[横尾 98b] 横尾, 平山：分散 breakout：反復改善型分散制約充足アルゴリズム, 情報処理学会論文誌, Vol. 31, No. 1, pp. 106–114 (1998).

[Yokoo 01] Yokoo, M.: *Distributed Constraint Satisfaction: Foundation of Cooperation in Multi-agent Systems*, Springer (2001).