人工知能学会誌 Vol. 10 No. 4 (Jul. 1995) 〔別刷〕

ショートノート

Research Notes

分散制約充足におけるエージェントの非集中的組織化

Organizing Agents in Decentralized Manner in Distributed Constraint Satisfaction

平山 勝敏*,† 山田 誠二*
Katsutoshi Hirayama Seiji Yamada
豊田 順一*
Jun'ichi Toyoda

* 大阪大学産業科学研究所
ISIR, Osaka University, Ibaraki, Osaka 567, Japan.

1994年6月7日 受理

Keywords: dynamic organizational structure, distributed constraint satisfaction problems, task distribution.

Summary

We have proposed LMO (Local Minimum driven Organizing) as a group forming mechanism for agents that solve Distributed Constraint Satisfaction Problems. It is summarized as follows. When an agent (A1) gets caught in a local minimum, 1) A1 sends its local CSP to the agent (A2) that shares a violated constraint, 2) A2 puts their CSP together and searches for all possible assignments with simple backtracking.

One flaw of LMO is that the cost of search grows exponentially with the number of CSP being put together. To cope with this flaw we extend our method as follows and introduce the dynamically weight adjusting strategy:

- We associate a weight with each constraint. The weights have 1 as their initial values;
- We measure the cost of instantiation as the sum of weights of violated constraints;
- Strategy: after putting CSP together, A2 sends neighbors the number of variables or the size of domain; neighbors reassign it for the weights of constraints being shared with A2.

The result of the experiment on distributed 3-coloring problem was that this strategy tended to protect against concentration of CSP and reduced the cost of search.

† 現在，神戸商船大学輸送情報システム工学科

1. はじめに

我々は，分散問題解決における組織形成の一手法としてLMO(Local Minimum driven Organizing)を提案した[平山 93, Hirayama 94b, 平山 95]．そこでは，分散制約充足問題(Distributed Constraint Satisfaction Problems: Distributed CSP)[横尾 92]を山登り法で並列に解くエージェントが，局所最適解に陥るごとに近傍の1エージェントと組織化するというアプローチがとられている．

LMOには，局所最適解をきっかけとし，それを解消するための組織形成という特徴がある．しかし，局所最適解が数多くあり，かつ特定のエージェントが繰り返し組織化した場合，大きな組織が形成され，組織内での問題解決コストが指数的に増加する．

本稿では，この問題の解決策として，問題解決中に制約の重みを動的に変えるというヒューリスティクス(動的重み調整)を提案し，その効果を実験的に示す．

2. 分散制約充足問題

Distributed CSPは，制約充足問題(Constraint Satisfaction Problems: CSP)における変数と制約が複数のエージェントに分散された問題と解釈できる．その定式化は[横尾 92]に詳しい．CSPでは，初期状態において1エージェントが全体の問題に見渡すことができるのに対し，Distributed CSPの特徴は，初期状態においてどのエージェントも全体の問題を知らない点である．各エージェントは独自の変数，値域，そして自分が持つ変数に関わる制約とその制約を共有するエージェント(近傍)のアドレスのみを知る．重要なことは，一つのCSPがあって，それを複数エージェントに分散した結果，上のような状況になったのではなく，そもそも上のような状況が出発点となることである．以下では，初期状態において各エージェントは一つの変数を持ち，かつ，制約は2変数間にのみ存在すると仮定する．

3. LMO

我々は，分散問題解決における組織形成の一手法としてLMO(Local Minimum driven Organizing)を提案した．その概略は，各エージェントが近傍と交渉し

ながら並列に山登り法を実行し，局所最適解に陥ったエージェントが矛盾制約を共有するエージェントと組織化するというものである．局所最適解の定義およびアルゴリズムの詳細は，[平山 95]を参照されたい．ここでは，以降の説明に必要な出登り法と組織化の概略を述べる．

（1） 山登り法

エージェントは，自分が持つ制約のうち，矛盾している制約の数を評価関数とする．近傍との交渉の結果，自分の変数の値（具体化）の変更が認められた場合には，制約矛盾数最小の値に具体化を変更する（Min-Conflicts ヒューリスティクス[Minton 92]）．もし，その候補が複数個あればランダムに一つ選択する．

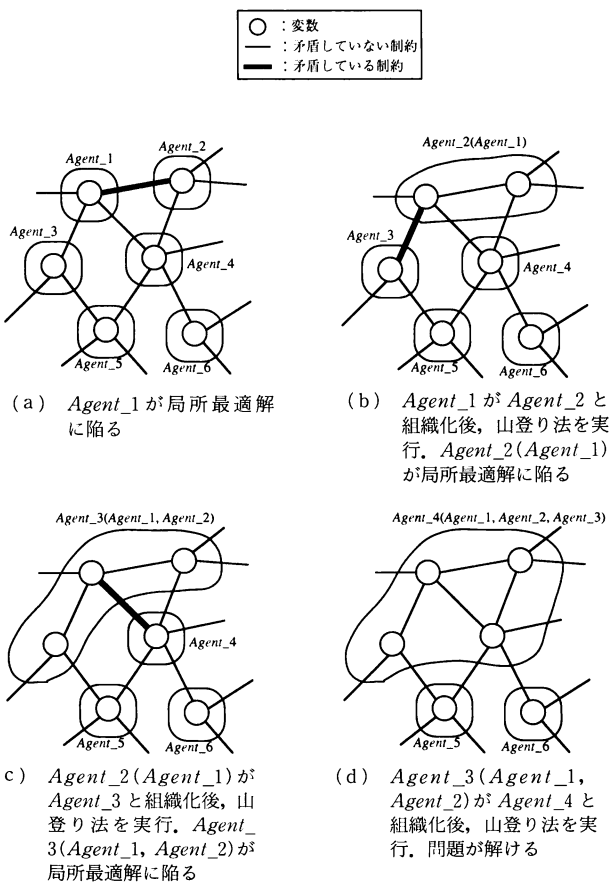
（2） 組織化

まず，局所最適解に陥ったエージェント（依頼エージェント）が，矛盾制約を共有するエージェントをランダムに一つ選び，自分の変数，値域，制約などを送信する．次に，受信したエージェント（更新エージェント）は，以下のように自分の変数，値域，制約を更新する．

- 変数：依頼エージェントと更新エージェントのそれぞれの変数の和集合を新たな変数とする．
- 値域：依頼エージェントと更新エージェントのそれぞれの変数と値域，および，両者が共有する制約で構成される CSP（局所 CSP）を単純バックトラッキング[Kumar 92]で全解探索し，得られた解すべてを新たな値域とする．
- 制約：依頼エージェントと更新エージェントのそれぞれの制約の和集合から，両者が共有する制約を除いた集合を新たな制約とする．

最後に，更新後の変数，値域，制約をもとに更新エージェントは制約矛盾数最小の値を求め，それを新たな具体化として近傍に送信する．ただし，ここでの近傍とは，更新後の制約を共有するエージェントのことである．

組織化後は，更新エージェントが山登り法を継続する．なお，上のように組織化の手続きを実行することを「依頼エージェントが更新エージェントと組織化す



（a） *Agent*_1 が局所最適解に陥る

（b） *Agent*_1 が *Agent*_2 と組織化後，山登り法を実行．*Agent*_2(*Agent*_1) が局所最適解に陥る

（c） *Agent*_2(*Agent*_1) が *Agent*_3 と組織化後，山登り法を実行．*Agent*_3(*Agent*_1, *Agent*_2) が局所最適解に陥る

（d） *Agent*_3(*Agent*_1, *Agent*_2) が *Agent*_4 と組織化後，山登り法を実行．問題が解ける

図1 特定のエージェントが繰り返し局所最適解に陥るようす

に重み2(変数の数)を送信し，近傍はこの組織との制約の重みを2にする．この結果，以降の山登り法において *Agent_3* と *Agent_4* は，この組織との制約を満たす傾向が強くなる．以下，図2(b)(c)(d)では，同様のことが繰り返される．

このように制約の重みを動的に調整することにより，大きな組織へのエージェントの集中をある程度抑えることができると考えられる．

5. 評　　価

5・1 評 価 方 法

ヒューリスティクスの効果を確かめるための実験を行う．次の三つの方法を比較した．

(**方法1**)　ヒューリスティクスなし

(**方法2**)　動的重み調整（重み＝変数の数）

(**方法3**)　動的重み調整（重み＝値域のサイズ）

取り上げる問題は，分散グラフ着色問題の $k=3, m=2n$(k：色の数，m：制約の数，n：変数の数)の場合である．グラフ着色問題に関して，この設定の連結グラフには，多くの局所最適解があることが報告されている[Minton 92]．実験では，$n=30$，60，90，120，150のそれぞれについて制約が異なる5種類の問題例をランダムに作成し*，各エージェントに変数一つとその値域，その変数に関する制約を割り当てる．また，各問題例につき100問の初期値をランダムに生成し，それぞれ各エージェントの変数の初期状態における具体化とする．つまり，各 n につき問題例5種類，初期値100問の計500問を解く．実験は，すべて並列処理用のシミュレータ上で行う．これは，エージェント間のメッセージ通信をシミュレートするためのもので，その構成については[平山 95]を参照されたい．

5・2 結果1：コスト

まず，各問について次の項目を測定し，それらの各 n ごとの500問平均を測定した．

- *check*：1エージェント当りの平均制約チェック回数．
- *max*：エージェントのサイズ(変数の数)の最大値．
- *message*：使用したメッセージ数．

実験結果を表1に示す．ただし，実験を妥当な時間で終わらせるために，1エージェントの値域更新時の制約チェック回数が40 000回を越えたときには処理を打ち切った．打ち切った場合のデータはその時点での値とし，*check* の欄の括弧内に500問中この制限以内に解けた問の割合を示す．

この実験に関する限りは，(方法2)(方法3)は，すべての項目において(方法1)よりも良くなっている．特に，(方法1)が非常に悪かった $n=90, 150$ の場合に大きく改善された．

5・3 結果2：組織のサイズ

次に，各問が解けた時点での組織のサイズ（変数の数）の分布を調べた．例えば，図1(d)で問題が解けたとすると，サイズ4の組織が一つ，図2(d)では，サイズ2の組織が三つ形成されたという．図3は，先の実験より得られた組織のサイズの分布を表すグラフで，$n=30$ の500問の総計を示している．他の場合($n=60, 90, 120, 150$)については省略するが，グラフの形は同じである．これから次の特徴を読み取ることができる．

- 大きな組織の数：(方法1)では，少数ながら大きな組織が形成されているが，(方法2)(方法3)ではそのような組織の数は非常に少ない．例えば，サイズ15以上の組織の数をすべての n について

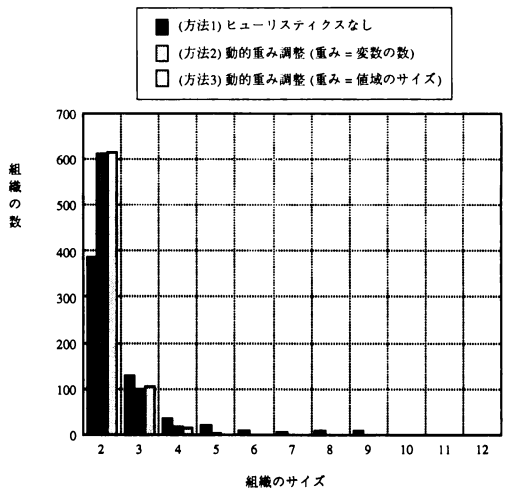


図3　組織形成のようす($n=30$ の場合)

表1　コスト

n	(方法1)			(方法2)			(方法3)		
	check	*max*	*message*	*check*	*max*	*message*	*check*	*max*	*message*
30	105.4	2.37	1203.4	53.7	1.82	1165.6	52.4	1.80	1161.8
60	98.1	3.26	2452.3	55.2	2.30	2355.8	53.2	2.27	2352.2
90	1600.1 (0.92)	5.88	5731.3	123.9	3.15	5105.7	109.4	3.02	5122.7
120	111.4 (0.99)	3.96	4644.8	50.4	2.58	4485.2	49.7	2.52	4482.9
150	3880.6 (0.80)	8.87	11925.3	917.8 (0.98)	4.36	10955.2	550.6 (0.99)	4.02	11331.5

* グラフの連結性はチェックしていない．

る」という．

本研究では，初期状態において，各エージェントは一つの変数を持つと仮定しているが，組織化後は更新エージェントが複数の変数を持つことになる．本稿では複数の変数を持つエージェントを特に「組織」と呼ぶが，以下では，特に必要がない場合には組織とエージェントを区別しない．また，組織のサイズとは，更新エージェントが持つ変数の数，あるいは値域のサイズを指す．

4. LMO の問題点とその改善法

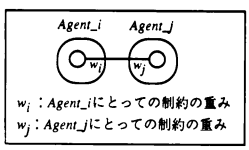
LMO では，更新エージェントが局所 CSP を全解探索して値域を更新する．このとき局所 CSP が大きいと全解探索のコストが非常に大きくなる．局所 CSP が大きくなる主な原因としては，1)依頼エージェントがサイズの大きいエージェントを更新エージェントとして選択する．2)特定のエージェントが繰り返し局所最適解に陥る，の 2 点が考えられる．図 1 は 2)の例である．

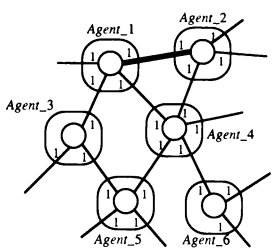
この問題に対して本稿では，問題解決中に制約の重みを動的に変えるというヒューリスティクス「動的重み調整」を提案する．従来の方法を次のように拡張する．

- 制約に重みを導入する．初期状態においてその重みは 1 とする．
- 矛盾する制約の重みの和を出登り法の評価関数とする．
- 組織化後，組織が近傍に重み w を送信し，近傍がその組織と共有する制約の重みを w にする．w の値は，組織が持つ変数の数，または値域のサイズとする．

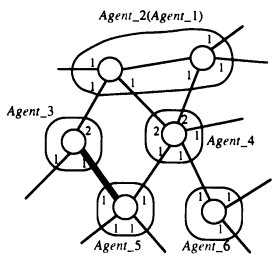
この拡張によりエージェントは，大きな組織と共有する制約をできるだけ満たすようになり，大きな組織と組織化する機会が減る．その結果，局所 CSP の増大が抑えられると予想される．このようすを図 2 に示す．

まず，図 2 (a)のようにすべての制約の重みが 1 に設定されているとして，*Agent*_1 が局所最適解に陥ったとする．すると，図 2 (b)のように *Agent*_1 が *Agent*_2 と組織化する．組織化後，*Agent*_2(*Agent*_1)が近傍

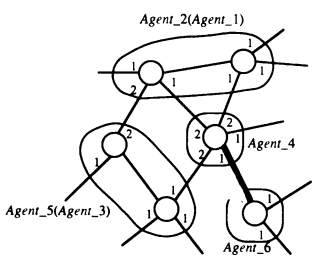




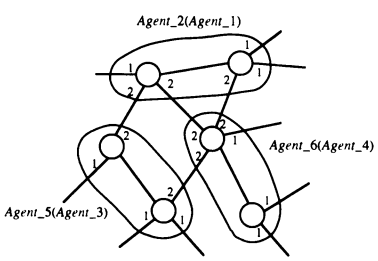
(a) *Agent*_1 が局所最適解に陥る



(b) *Agent*_1 が *Agent*_2 と組織化後，出登り法を実行．*Agent*_3 が局所最適解に陥る



(c) *Agent*_3 が *Agent*_5 と組織化後，山登り法を実行．*Agent*_4 が局所最適解に陥る



(d) *Agent*_4 が *Agent*_6 と組織化後，山登り法を実行．問題が解ける

図 2 動的重み調整

表2　組織の総数

n	(方法1)	(方法2)	(方法3)
30	602	733	736
60	1361	1609	1613
90	2658	3686	3762
120	2425	2828	2848
150	4822	7312	7587

調べてみると，(方法1)では115，(方法2)では6，(方法3)では2である．

- 小さな組織の数：(方法1)に比べて，(方法2)(方法3)には小さな組織が数多く形成されている．例えば，サイズ2の組織の数をすべてのnについて調べてみると，(方法1)では7 050，(方法2)では12 273，(方法3)では12 347である．
- 組織の変数：各nについて，形成された組織の総数を表2に示す．明らかにどのnについても，組織の数は(方法1)よりも(方法2)(方法3)のほうが多くなっている．

以上から，各方法の組織形成について次のことがいえる．すなわち，(方法1)では，少数の大きな組織が形成されるが，組織の総数は少ない．(方法2)(方法3)では，大きな組織は形成されないが，多数の小さな組織が形成され，組織の総数は多い．

6. 考　　察

動的重み調整では，大きなエージェントAの近傍はAとの制約をできるだけ壊さずに，A以外のエージェントとの制約を壊すことになる．この結果，Aの具体化は変更されにくくなる．もし，ここでAが誤った値(Distributed CSPの解の一部でない値)に具体化していたとすると，その値が変更されないままAの近傍にむだな処理を強いることになる．よって，大きなエージェントAが誤った値に具体化することが多いと，AのサイズだけをもとにAの近傍が重みを調整する方法はかえって非効率になると考えられる．この問題を解決するためには，重みの設定にエージェントのサイズ以外の要因，例えば具体化の正しさなどを反映させる必要がある．

また，局所CSPの増大という問題に対し，動的重み調整以外にもいくつかの方法が考えられる．例えば，更新エージェントの値域更新を全解探索ではなく一解探索とし，以降，値変更の要求があるたびに順次解を探索するという方法(一解探索)，依頼エージェントが更新エージェントを選択する基準に更新エージェントのサイズを反映させる方法(更新エージェントの選択)である．しかし，これら二つについて5•1節と同じ設定のもとで実験した結果，動的重み調整ほどの改善は見られなかった[平山 94a]．

7. ま　と　め

本稿では，局所CSPの増大という問題を解決するために，エージェントが問題解決中に動的に制約の重みを変えるというヒューリスティクスを提案し，それが有効であることを確認した．さらに，このヒューリスティクスが有効に働くクラスについて考察し，重みの設定方法についてはまだ未解決であることを述べた．今後の課題は，重みの設定に対する有効なアプローチを探すことにある．

◇参考文献◇

[平山 93]　平山勝敏，山田誠二，豊田順一：山登り法を用いた分散制約充足における組織化，情処学会人工知能研資，93-AI-90, pp. 23-32 (1993).

[平山 94a]　平山勝敏，山田誠二，豊田順一：分散制約充足における組織化の負荷分散，人工知能学会全大(第8回)，pp. 249-252 (1994).

[Hirayama 94b]　Hirayama, K., Yamada, S. and Toyoda, J.: A Dynamic Organization in Distributed Constraint Satisfaction, *AAAI-94: Student Abstract*, pp. 1456 (1994).

[平山 95]　平山勝敏，山田誠二，豊田順一：山登り法を用いた分散制約充足における組織化，人工知能学会誌，Vol. 10, No. 1, pp. 80-87 (1995).

[Kumar 92]　Kumar, V.: Algorithms for Constraint Satisfaction Problems: A Survey, *AI Magazine*, Vol. 13, No. 1, pp. 32-44 (1992).

[Minton 92]　Minton, S., Johnston, M. D., Philips, A. B. and Laird, P.: Minimizing Conflicts: A Heuristic Repair Method for Constraint Satisfaction and Scheduling Problems, *Artif. Intell.*, Vol. 58, pp. 161-205 (1992).

[横尾 92]　横尾　真，Durfee, E. H.，石田　亨，桑原和宏：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，信学論，Vol. J-75 D-I, No. 8, pp. 740-713 (1992).

〔担当編集委員：堂下修司，査読者：石田　亨〕

著 者 紹 介

平山　勝敏(正会員)

1990 年大阪大学基礎工学部卒業．1992 年同大学院修士課程修了．現在，同大学院博士課程存学中．分散 AI，制約充足などに興味を持つ．

山田　誠二(正会員)

1984 年大阪大学基礎工学部卒業．1989 年同大学院博士課程修了．同年，基礎工学部システム工学科助手．1991 年より大阪大学産業科学研究所講師，現在に至る．工学博士．人工知能，特に，効率化学習，即応プラニング，マルチエージェント系に興味を持つ．情報処理学会，日本認知科学会，日本ソフトウェア科学会，AAAI，IEEE 各会員．

豊田　順一(正会員)

1961 年大阪大学工学部卒業．1966 年同大学院博士課程単位取得退学．同年，大阪大学基礎工学部助手．1969 年助教授．1982 年大阪大学産業科学研究所教授．工学博士．現在，概念の形成，Visual fidelity，マニュアルのわかりにくさに関する研究に従事．情報処理学会，電子情報通信学会，日本認知科学会各会員．