知能と複雑系　132－12
(2003. 3. 14)

情報収集のための分散タスク割り当て

平山勝敏† 北村泰彦††

† 神戸商船大学　〒658–0022 神戸市東灘区深江南町 5–1–1
†† 大阪市立大学大学院工学研究科　〒558–8585 大阪市住吉区杉本町 3–3–138
E-mail: †hirayama@ti.kshosen.ac.jp, ††kitamura@info.eng.osaka-cu.ac.jp

あらまし　本論文では，マルチエージェントによる情報収集統合モデルにおける情報源監視タスク割り当て問題の解法について述べる．このモデルは情報統合エージェント群と情報収集エージェント群からなり，情報統合エージェントはユーザの情報収集要求に応じて，監視すべき情報源を情報収集エージェントに割り当てる．その際，情報統合エージェントは，ユーザの情報収集要求や情報収集エージェントの資源制約などを考慮して適切な割り当てを求める必要があり，しかも，そのような割り当てを，ユーザの情報収集要求をできるだけ外部に公開せずに求めたい．本論文では，このような情報源監視タスク割り当て問題を分散 SAT として定式化し，汎用の分散 SAT アルゴリズムで解くことにより解決することを目指す．

キーワード　タスク割り当て，分散制約充足問題，SAT

Distributed Task Assignment for Information Gathering

Katsutoshi HIRAYAMA† and Yasuhiko KITAMURA††

† Kobe University of Mercantile Marine　Fukaeminami-machi 5–1–1, Higashinada-ku, Kobe, 658–0022 Japan
†† Graduate School of Engineering, Osaka City University　Sugimoto-cho 3–3–138, Sumiyoshi-ku, Osaka, 558–8585, Japan
E-mail: †hirayama@ti.kshosen.ac.jp, ††kitamura@info.eng.osaka-cu.ac.jp

Abstract　This paper describes a method to solve the distributed task assignment problem on a simple model of multi-agent information gathering. This model consists of two types of agents: *Information Integration agents* (I-agents) and *Information Gathering agents* (G-agents). I-agents assign tasks of watching some information sources to G-agents and G-agents notify I-agents of renewal of information sources that they have been watching. In this model, I-agents need to assign watching tasks so that the assignment meets not only users' requirements but also resource constraints imposed upon G-agents. Also, it is desirable that I-agents can obtain such an assignment without revealing users' requirements each other because such requirements may include some private information. In this paper, we encode this assignment problem as distributed SAT and solve it using a general-purpose distributed SAT algorithm.

Key words　Task assignment, Distributed CSP, SAT

1. まえがき

インターネットをベースとした Web は今や地球規模に広がり，情報を発信するための基本的な道具として広く利用されている．Web の特徴は，集中的な管理機構なしに膨大な数の情報源がボトムアップに構築され，それらが互いに連携しながら，あたかも一つの自律分散型データベースのように振舞う点である．しかし利用者にとっては，そのような大規模でしかも頻繁に更新される情報源から効率よく有用な情報を発見することは一般に困難である．本研究では，Web のような動的で大規模な情報源から効率よく有用な情報を発見する手段の一つとして，マルチエージェントによる情報収集・監視・統合モデルを構築し，それに基づいたシステムの設計および基本的なアルゴリズムの設計を目指す．

Web 情報源の情報を収集、監視、統合するための一つの枠組みとして図 1 に示すようなマルチエージェントによる情報収集・監視・統合モデルを構築する。このモデルでは、情報収集エージェントと情報統合エージェントという 2 種類のエージェントが存在する。このモデルでの情報の収集・監視・統合の手順の概略は次の通りである。

監視タスクの割り当て　ユーザからの情報収集要求を受けて，情報統合エージェントは情報収集エージェントに監視タスクを

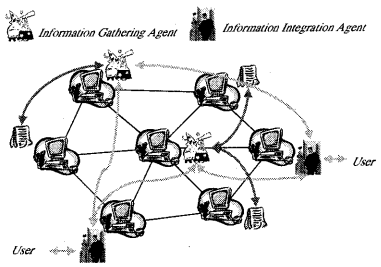

図 1　The Model

割り当てる．情報統合エージェントは，ユーザの情報収集要求や情報収集エージェントの資源制約を考慮して適切な割り当てを求める．

情報源の監視　監視タスクを割り当てられた情報収集エージェントは，自分の監視スケジュールに対応する情報源を加え、以後、そのスケジュールに従ってその情報源を監視する。

情報源の内容変更通知　監視している情報源の内容に変更があった場合には、その旨を対応する情報統合エージェントに通知する。

情報源の内容変更に伴う処理の実行　情報収集エージェントから内容変更を通知された情報統合エージェントはそれに伴った処理（ユーザに通知する、別の情報源に切り替える等）を実行する。

このモデルで，個々の情報統合エージェントが独立に監視タスクを情報収集エージェントに割り当てると，場合によっては特定の情報収集エージェントの資源制約が満たされず，タスクの実行が不可能になる．このような事態を避けるための一つの方法として，情報統合エージェントがユーザの情報収集要求を一箇所に集め，各情報収集エージェントの資源制約を満たすようなタスクの割り当てを集中型のアルゴリズムにより求めることが考えられる．しかし，この方法では，個々のユーザの情報収集要求に関する情報が情報統合エージェント以外にも漏れることになり，セキュリティやプライバシーの面で適当とはいえない．そこで，本研究では，ユーザの情報収集要求を集めることなく，情報統合エージェントどうしで必要な情報だけを交換することにより，監視タスクの適切な割り当てを求めることを目指す．

情報を一箇所に集めずに分散した状態でタスク割り当てを求める最も簡単な方法は，それぞれの情報統合エージェントが，局所的な情報に基づく割り当て規則を使ってグリーディにタスクを情報収集エージェントに割り当てることである．しかし，このような方法では，いかに巧妙な割り当て規則を考案するかが重要であるが，一般には，そのような規則を考案することは困難であり，しかも，情報統合エージェントが空間的に分散している場合には特に困難である．

そこで本研究では，情報統合エージェントによるタスク割り当て問題を分散 SAT として定式化し，汎用の分散 SAT アルゴリズムを用いて適切な割り当てを求めるというアプローチをとる．

以下，本論文の構成は次のとおりである．2. 節では，SAT と分散 SAT の定義を述べる．3. 節では，情報統合エージェントによる監視タスク割り当て問題を分散 SAT として定式化するための方法を示す．4. 節では，分散 SAT を解くアルゴリズムである MULTI-DB について説明し，5. 節で簡単な問題例での評価結果を示す．最後に，6. 節でまとめと今後の課題を述べる．

2. 分散 SAT

充足可能性判定問題（SAT）とは連言標準形で記述された命題論理式に対して，それを真にするような命題変数（以降，単に変数）への真偽値（T または F）の割り当てが存在するかどうかを判定する問題である．近年，人工知能における制約充足問題やプランニングなどの分野で，高速な SAT ソルバーの開発，問題の性質の解析，組み合わせ問題の SAT によるコード化などを主なテーマとして，SAT は多くの研究者の注目を集めている．

連言標準形の命題論理式は節の論理積からなり，節はリテラル（変数あるいはその否定）の論理和からなる．連言標準形の命題論理式の例を以下に示す．

$$(x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2) \wedge (x_3 \vee x_4) \wedge (\neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_3) \wedge (\neg x_2 \vee \neg x_4)$$

この式は例えば $(x_1, x_2, x_3, x_4) = (\mathrm{T}, \mathrm{F}, \mathrm{F}, \mathrm{T})$ のとき真となるため，この割り当てが一つの解となる．ちなみに，この式は，1 つの無向枝で接続された 2 つの節点 n_1, n_2 からなるグラフに対する 2 色問題（隣接する節点が互いに異なる色になるようにグラフ全体を白と黒の 2 色で塗り分ける問題）を SAT として記述したものである．ここで，x_1（x_2）は節点 n_1 が白（黒）のときに T となる変数である．一方，x_3（x_4）は節点 n_2 が白（黒）のときに T となる変数である．

分散 SAT は，SAT における変数や節が複数のエージェントに分散された問題である．ここでは，それぞれ次のように分散されていると仮定する．

- 変数は複数のエージェントに分割される．すなわち，同じ変数が複数のエージェントに属するということはない．
- 節は，それが関係する変数をもつすべてのエージェントに割り当てられる．すなわち，ある節に含まれる複数の変数が複数の異なるエージェントに割り当てられていた場合，その節はそれらのエージェントすべてに割り当てられる．

なお，分散 SAT の解とは，エージェントに分散されているすべての節を真にするような変数への真偽値の割り当てのことである．

例えば，前述の連言標準形の命題論理式の例で，4 つの変数 $(x_1, \ldots, x_4)$ と 6 つの節（左から $C_1, \ldots, C_6$ とよぶ）が，図 2 のように 2 つのエージェントに分散され，エージェント 1 は変数 x_1, x_2 と節 C_1, C_2, C_5, C_6 を，エージェント 2 は変数 x_3, x_4 と節 C_3, C_4, C_5, C_6 をもつとする．ここで，節 C_5, C_6 は両方のエー

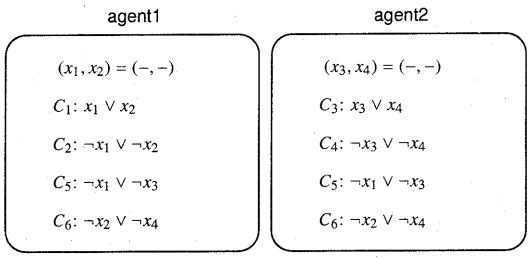

図 2　Example of DisSAT

ジェントに割り当てられているが，それは，それらが両方のエージェントがもつ変数に関わっているからである．そのような節のことをエージェント間節とよぶ．一方，節 C_1, C_2, C_3, C_4 は 1 エージェント内の変数にしか関わっていないが，そのような節のことをエージェント内節とよぶ．また，あるエージェントにとってエージェント間節を通して互いに関係のあるエージェントの集合のことをそのエージェントの近傍とよぶ．

3. 定 式 化

情報統合エージェントによる監視タスク割り当て問題は次のようにして分散 SAT として定式化することができる．

エージェント　情報統合エージェントを分散 SAT を構成するエージェントとする．

変数　情報統合エージェント a が情報収集エージェント x に情報源 i を割り当てることを論理変数 x_i^a で表す．すなわち，x_i^a が真のときかつそのときのみ，a は x に i を割り当てる．

節集合 1　情報統合エージェントがユーザの情報収集要求を満たすために情報収集エージェント群に監視してもらいたい情報源を節集合 1 として記述する．例えば，情報統合エージェント a が情報収集エージェント x と y に情報源 1 と 2 を監視してもらいたい場合（どちらが 1 でも 2 でもよい），節集合 1 は $(x_1^a \vee y_1^a) \wedge (x_2^a \vee y_2^a)$ を含む．

節集合 2　1 つの情報収集エージェントに一度に割り当てることができる情報源の数の上限値を設定し，その数が上限値以下の場合には真，上限値を超える場合には偽を返すような節集合 2 を追加する．例えば，情報統合エージェント a が，情報収集エージェント x は情報源 1, 2, 3 を監視できるが同時に監視できるのは 2 つ以下であることを知っている場合，節集合 2 は，$\neg x_1^a \vee \neg x_2^a \vee \neg x_3^a$ を含む．

節集合 3　ある情報収集エージェントにどの情報源を割り当てるかということに関して複数の情報統合エージェント間で合意する必要があるため，そのような節集合 3 を追加する．例えば，情報統合エージェント a と b の間で，情報収集エージェント x に対して情報源 1 を割り当てることを合意するためには，節集合 3 は $(\neg x_1^a \vee x_1^b) \wedge (x_1^a \vee \neg x_1^b)$ を含む必要がある．

簡単な例題を示す．2 つの情報統合エージェント a, b，2 つの情報収集エージェント x, y がいて，それぞれの情報収集エージェントは情報源 1, 2, 3 のうち最大 2 つの情報源を監視できるものとする．情報統合エージェント a は，ユーザの情報収集要求を満たすために，情報収集エージェント x と y に情報源 1 と 2 を監視してもらいたいとする．一方，情報統合エージェント b は，情報収集エージェント x と y に情報源 1 と 3 を監視してもらいたいとする．この場合，情報統合エージェント a と b はそれぞれ次のような SAT をもつことになる．なお，便宜上，節集合 1，2，3 に分けているが，実際にはそれらの積が満たすべき節集合となる．

情報統合エージェント a　：

変数　：$\{x_1^a, x_2^a, x_3^a, y_1^a, y_2^a, y_3^a\}$

節集合 1　：$(x_1^a \vee y_1^a) \wedge (x_2^a \vee y_2^a)$

節集合 2　：$(\neg x_1^a \vee \neg x_2^a \vee \neg x_3^a) \wedge (\neg y_1^a \vee \neg y_2^a \vee \neg y_3^a)$

節集合 3　：$(\neg x_1^a \vee x_1^b) \wedge (x_1^a \vee \neg x_1^b) \wedge (\neg x_2^a \vee x_2^b) \wedge (x_2^a \vee \neg x_2^b) \wedge (\neg x_3^a \vee x_3^b) \wedge (x_3^a \vee \neg x_3^b) \wedge (\neg y_1^a \vee y_1^b) \wedge (y_1^a \vee \neg y_1^b) \wedge (\neg y_2^a \vee y_2^b) \wedge (y_2^a \vee \neg y_2^b) \wedge (\neg y_3^a \vee y_3^b) \wedge (y_3^a \vee \neg y_3^b)$

情報統合エージェント b　：

変数　：$\{x_1^b, x_2^b, x_3^b, y_1^b, y_2^b, y_3^b\}$

節集合 1　：$(x_1^b \vee y_1^b) \wedge (x_3^b \vee y_3^b)$

節集合 2　：$(\neg x_1^b \vee \neg x_2^b \vee \neg x_3^b) \wedge (\neg y_1^b \vee \neg y_2^b \vee \neg y_3^b)$

節集合 3　：$(\neg x_1^a \vee x_1^b) \wedge (x_1^a \vee \neg x_1^b) \wedge (\neg x_2^a \vee x_2^b) \wedge (x_2^a \vee \neg x_2^b) \wedge (\neg x_3^a \vee x_3^b) \wedge (x_3^a \vee \neg x_3^b) \wedge (\neg y_1^a \vee y_1^b) \wedge (y_1^a \vee \neg y_1^b) \wedge (\neg y_2^a \vee y_2^b) \wedge (y_2^a \vee \neg y_2^b) \wedge (\neg y_3^a \vee y_3^b) \wedge (y_3^a \vee \neg y_3^b)$

各情報統合エージェントがもつことになる問題のサイズについて述べる．まず，変数の数については，各情報統合エージェントが情報源の監視を依頼する情報収集エージェントの数を n，各情報収集エージェントが監視できる情報源の潜在数を s とすると，各情報統合エージェントがもつ変数の数は ns となる．節集合 1 については，n 個の情報収集エージェントに対し特定の t 個の情報源を任意に割り当てる場合，生成される節の数は t 個となる．任意に割り当てるのではなく，特に情報収集エージェントを指定して割り当てる場合にもやはり t 個となる（任意に割り当てる場合よりも節に含まれるリテラルの数が少なくなる）．節集合 2 については，各情報収集エージェントが一度に監視することができる情報源の数の上限値を u とすると，生成される節の数は $n \cdot {}_sC_{u+1}$ となる．節集合 3 については，情報統合エージェントの数を m とすると，生成される節の数は $ns \cdot {}_mC_2$ となる．

4. アルゴリズム

前節の方法により情報統合エージェントによるタスク割り当て問題が分散 SAT として定式化された．本論文では，これを汎用の分散 SAT アルゴリズムで解くことにする．利用するアルゴリズムは Multi-DB [1] である．これは，1 エージェントが複数変数をもつ分散 SAT を解くことができる反復改善型の分散 SAT アルゴリズムであり，前節の方法により得られた分散 SAT に対し，特別な変更を加えることなくそのまま適用することが可能である．

Multi-DB の基本的な動作は，全エージェントが互いに関連する情報を交換しながら並行して局所探索法を実行し，変数に対する初期の値割り当てが解に近づくように繰り返し改善していくというものである．手続きの詳細は [1] に譲り，ここでは

各エージェントの動作の概略を図3に示す．ステップの概要は次の通りである．

エージェントは変数の初期値を変えて *MAXTRIES* で指定された回数だけ試行を繰り返す．各試行では，エージェントは自分の変数の値をランダムに設定し（ステップ02），*MAXROUNDS* で指定された回数だけ（ステップ04）から（ステップ24）の手続きを繰り返す．以下，（ステップ04）から（ステップ24）の概要を説明する．

（ステップ04）から（ステップ06）： 近傍と変数の値を交換した後，現在の状態の評価値（違反している節の重み和）を求め，さらに，その評価値を改善できる値反転の集合（値を反転する変数の集合）である *PossFlips* を局所探索法により求める．その後，近傍と *PossFlips* を交換する．

（ステップ08）： 近傍と自分の範囲で違反している節が存在しない場合，分散ブレイクアウト法（DB）と同様の終了判定処理[9]を行う．

（ステップ11）： 近傍と自分の範囲で違反している節が存在し，かつ，同範囲で *PossFlips* が存在しない場合は擬似局所最適[9]に相当する．この場合，DBと同様にその時点で違反している節の重みを上げることにより，擬似局所最適から脱出しようとする．

（ステップ13）から（ステップ17）： 近傍と自分の範囲で違反している節が存在し，かつ，同範囲で2つ以上のエージェントが *PossFlips* をもつ場合には注意が必要である．というのは，それらの値をすべて反転した結果，その効果がいわゆる「干渉」を起こし，それぞれのエージェントが意図していた評価値の改善が得られない場合があるからである．MULTI-DBでは，エージェントが近傍と事前に *PossFlips* を交換して次の状態を調べ，次の状態で新しく違反となる節をすべて求める．そして，その各節について，違反の原因となる *PossFlips* をもつエージェントが自分を含めて複数存在し，かつ，その中で自分の評価値の改善度が最小ならば（改善度が同じ場合は識別子の値が大きければ），その違反の原因となっている自分の変数の値反転を *PossFlips* から削除する．なお，そのような値反転が複数個ある場合にはランダムに1つ選んで削除する．

（ステップ18）から（ステップ22）： （ステップ13）から（ステップ17）の結果，自分の *PossFlips* で削除された変数がない場合は，*PossFlips* 内のどの変数の値を反転しても，予定外の新しい違反は起こらないため，*PossFlips* のすべての値を反転することができる．一方，削除された変数がある場合には，*PossFlips* 内の残された変数の値を反転する．ただし，残された変数の値を当初の予定通り反転させても無意味である．なぜなら，当初の *PossFlips* が評価値を改善できるのはそれ全体としてであり，その部分に同じ効果があるとはいえないからである．よって，残された変数だけを用いて再度局所探索法を実行し，評価値を改善できるようにそれぞれの値を決定する．

5. 評　　価

情報統合エージェントによる監視タスク割り当て問題を分散SATとして定式化し，MULTI-DBを用いて解くというアプローチの可能性を評価するために，シミュレータによる簡単な評価実験を行った．

このシミュレータは同期型の分散システムをシミュレートしたもので，そこでは，すべてのエージェントが一斉に，直前に発せられたメッセージの受信，内部計算，他エージェントへのメッセージの送信，という一連の処理（サイクルとよぶ）を繰り返す．このシミュレータ上で，MULTI-DBを実現し，その通信コストと計算コストとして，次のcyclesとflipsを測定する．

cyclesは，解を発見するまでに要したシミュレータ上でのサイクル数である．同期型の分散システムでは，各エージェントは1サイクルごとに他のエージェントとメッセージを送受信するため，このサイクル数をアルゴリズムの通信コストと見なすことができる．

一方，flipsは，各サイクルの内部計算においてエージェントが行った変数選択の回数の最大値を，解を発見するまでの全サイクルに渡って合計したものである．エージェントが行った変数選択の回数とは，MULTI-DBの場合，局所探索法における変数の（仮想的な）値反転回数に相当する．同期型の分散システムでは，各サイクルの内部計算において変数選択の回数が最大となるエージェントが，そのサイクルでの計算のボトルネックとなる．そこで，各サイクルごとにボトルネックとなるエージェントを1つ定め，そのようなボトルネックとなるエージェントの計算コスト（変数選択の回数）を，解を発見するまでの全サイクルに渡って合計したものをアルゴリズムの計算コストと見なす．

評価用の問題として2種類の簡単な問題例を用意した．それらの特徴を以下にまとめる．

問題例1　：

- 情報統合エージェント：$\{a, b\}$
- 各情報統合エージェントのタスク割り当て要求
  - a は情報源1と2を監視する．
  - b は情報源2と3を監視する．
- 情報収集エージェント：$\{x_1, x_2\}$
- 各情報収集エージェントが監視可能な情報源：$\{1, 2, 3\}$
- 各情報収集エージェントが一度に監視することができる情報源の数の上限値：2個

問題例2　：

- 情報統合エージェント：$\{a, b, c\}$
- 各情報統合エージェントのタスク割り当て要求
  - a は情報源1, 2, 3を監視する．
  - b は情報源2, 3, 4, 5を監視する．
  - c は情報源4, 5, 6を監視する．
- 情報収集エージェント：$\{x_1, x_2, x_3\}$
- 各情報収集エージェントが監視可能な情報源
  - x_1 は情報源1, 2, 3, 4
  - x_2 は情報源1, 2, 5, 6
  - x_3 は情報源3, 4, 5, 6
- 各情報収集エージェントが一度に監視することができる情報源の数の上限値：2個

問題例3　：

```
(01) for t := 1 to MAXTRIES do
(02)   set variable values randomly;
(03)   for r := 1 to MAXROUNDS do
(04)     exchange variable values with neighbors;
(05)     perform local search for PossFlips;
(06)     exchange PossFlips with neighbors;
(07)     if there is no violated clause among a and its neighbors then
(08)       perform the termination detection procedure;
(09)     else
(10)       if there is no PossFlips among a and its neighbors then
(11)         increase weights of violated clauses;
(12)       else
(13)         for each newly violated clause at the next possible state do
(14)           if (the violation would be caused by at least two agents including a)
                ∧ (a's PossFlips would have the least improve among them) then
(15)             withdraw one of a's PossFlips that would cause the violation;
(16)           end if;
(17)         end do;
(18)         if no flip of a's PossFlips is withdrawn then
(19)           perform all the flips in PossFlips;
(20)         else
(21)           perform local search again for the background flips;
(22)           perform the background flips;
(23)         end if;
(24)       end if;
(25)     end if;
(26)   end do;
(27) end do;
```

図 3　Sketch of Multi-DB

- 情報統合エージェント：$\{a, b, c\}$
- 各情報統合エージェントのタスク割り当て要求
  - a は情報源 1 と 2 を監視する．
  - b は情報源 3 と 4 を監視する．
  - c は情報源 5 と 6 を監視する．
- 情報収集エージェント：$\{x_1, \ldots, x_6\}$
- 各情報収集エージェントが監視可能な情報源：$\{1, 2, 3, 4, 5, 6\}$
- 各情報収集エージェントが一度に監視することができる情報源の数の上限値：1 個

各エージェントがもつ変数と節の数は，問題例 1 の場合はそれぞれ 6 個，16 個，問題例 2 の場合は 12 個，64 個（情報統合エージェント a と c は 63 個），問題例 3 の場合は 36 個，236 個となる．

表 1 に，各問題例に対する実行結果を示す．なお，問題例 1 については変数の初期値をランダムに 20 通り，問題例 2 と 3 についてはランダムに 50 通り生成し，表 1 には，cycles と flips のそれぞれについて平均値，中央値，標準偏差を示している．また，Multi-DB のパラメータの設定は，*MAXTRIES*=1, *MAXROUNDS*=1000 とし，各エージェントによる局所探索法には tabu リスト付の WalkSAT [6] を用いた（noise=0.3, tabu length=5）

この結果からわかる通りこの程度の規模の問題に対しては，Multi-DB は十分な性能を示している．特に問題例 3 の場合，全体の SAT のサイズは，変数の総数は 108 個，節の総数は 492 個（複数エージェント間で重複しているものを省く）と比較的規模が大きいにも関わらず，それほどコストは大きくない．Multi-DB が効率の良いアルゴリズムであることは過去の実験により確認されているが，それ以外に考えられる理由としては，問題例 3 の場合，492 個の節のうち 486 個までは 2 つのリテラルしか含んでいないため，実質的にはクラス P である 2-SAT に近い性質をもつためであると考えられる．

今回は分散 SAT へのコード化を手作業で行ったため，ごく簡単な問題例でしか評価できていない．今後は監視タスク割り当て問題を自動的に分散 SAT にコード化することにより実験の効率化をはかり，より複雑な問題例で評価する予定である．

6. まとめと今後の課題

本論文では，マルチエージェントによる情報収集統合モデルにおける監視タスク割り当て問題を分散 SAT として定式化する方法を示した．また，既存の分散 SAT アルゴリズムである Multi-DB を用いて実際に簡単な例題を解き，本アプローチの評価を行った．今後の課題は，さらに複雑な割り当て問題で本アプローチの評価を行うこと，既存の他の分散制約充足アルゴリズムを使って評価を行うこと等が挙げられる．

文　献

[1] K. Hirayama and M. Yokoo. Local search for distributed SAT with complex local problems. In *Proceedings of the First International Joint Conference on Autonomous Agents & Multi-Agent Systems*,

表 1 Results of MULTI-DB for example 1, 2, and 3

	cycles			flips		
	mean	median	stddev.	mean	median	stddev.
ex.1	6.0	6.0	1.41	21.0	20.0	6.24
ex.2	32.7	30.0	18.5	167.3	160.5	96.7
ex.3	89.2	90.0	48.0	461.3	470.5	246.3

pages 1199–1206, 2002.

[2] H. A. Kautz and B. Selman. Pushing the envelope: Planning, propositional logic, and stochastic search. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 1194–1201, 1996.

[3] D. Mitchell, B. Selman, and H. Levesque. Hard and easy distributions of SAT problems. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 459–465, 1992.

[4] P. J. Modi, H. Jung, M. Tambe, W.-M. Shen, and S. Kulkarni. A dynamic distributed constraint satisfaction approach to resource allocation. In *Proceedings of the Seventh International Conference on Principles and Practice of Constraint Programming*, pages 685–700, 2001.

[5] P. Morris. The breakout method for escaping from local minima. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 40–45, 1993.

[6] B. Selman, H. A. Kautz, and B. Cohen. Noise strategies for improving local search. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 337–343, 1994.

[7] B. Selman, H. Levesque, and D. Mitchell. A new method for solving hard satisfiability problems. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 440–446, 1992.

[8] M. Yokoo, E. H. Durfee, T. Ishida, and K. Kuwabara. The distributed constraint satisfaction problem: formalization and algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 10(5):673–685, 1998.

[9] M. Yokoo and K. Hirayama. Distributed breakout algorithm for solving distributed constraint satisfaction problems. In *Proceedings of the Second International Conference on Multi-Agent Systems*, pages 401–408, 1996.

[10] M. Yokoo and K. Hirayama. Distributed constraint satisfaction algorithm for complex local problems. In *Proceedings of the Third International Conference on Multi-Agent Systems*, pages 372–379, 1998.

[11] W. Zhang and Z. Xing. Distributed breakout vs. distributed stochastic: A comparative evaluation on scan scheduling. In *Proceedings of the Third International Workshop on Distributed Constraint Reasoning*, pages 192–201, 2002.