1F3-3

報酬配分に基づく強化学習を用いた効率的なチーム編成手法の提案

Effective coalition formation using reinforcement learning based on the utility distribution strategies

浜田　大[*1]　　菅原　俊治[*1]
Dai Hamada　Toshiharu Sugawara

[*1]早稲田大学大学院基幹理工学研究科
Fundamental Science and Engineering, Waseda University

We propose a coalition formation method in task oriented domains in multi-agent systems and experimentally evaluate it in this paper. Recently, service requests on computer networks have rapidly been increasing, and thus the load in such a system also becomes heavy. Therefore, the issues of effective and secure coalition formation attract our interest. We generalize the conventional coalition formation method in task oriented domain to archive more effects coalition formation using reinforcement learning. We introduce three parameters to agents : greedy degree, reward allocation ratio, and expectation of proposal acceptance and proposal model in which these parameters are learned in all the agents and all the agents decide whether they form a new team or they join one of proposal teams. Our experiment shows that the number of tasks executed by generated teams increase by 30% compared with the technique that agents do not learning.

1. 序論

近年，計算機ネットワーク上におけるサービスリクエストが急激に増加しており，それに伴うシステム内の負荷も増大している [1]．そのため，複数の計算機による協調負荷分散の手法に注目が集まっている．一般に協調負荷分散において，サービス実現のためのタスクは複数のサブタスクから構成され，それらを全て処理することでサービスが実現する．たとえ 1 つのサブタスクが未処理あるいは遅延するだけでもサービスの提供不可，または遅延として現れるため，全てのサブタスクを適切な能力と機能を持ったソフトウェアエージェントに割り当て，それらがグループとしてこれを処理する必要がある．このようなエージェントのグループを決める問題をチーム編成と呼ぶ．チーム編成問題は，この他にもアドホックネットワークのノード選択や広域の探索問題の抽象化と考えられ，多くの研究がなされている．

このような背景のもと，エージェントに自律的な学習を行わせ，チーム編成を効率化する研究が存在する．例えば [2] では，チーム編成の成功率を高めるため，過去に送受信したチーム編成の提案の類似度に基づいたエージェントの学習アルゴリズムを提案した．しかし，このモデルではシステム内に複数種類のタスクが無限に存在し，各々のエージェントはそれらを自由に選べる設定になっており，エージェントのスキルや報酬の値によっては要求されても処理されないタスクが生じる可能性があるため，現実のモデルとは必ずしも合致しない．また手法の性能指標として，チーム編成の成功率を確認するための実験は行われているが，実際にこのチーム編成手法を用いてシステムを運用した場合の効率化については議論されていない．

本研究では [2] のモデルと異なり，エージェントは環境から要求されたタスクを順に処理するとともに，処理の結果配分される報酬の配分割合やその期待値を学習・適応させる．具体的には，エージェントには報酬の配分を決める欲張り度，提案したチームへの参加受理の度合いを示す提案受託期待度，チームへ参加したときに受ける報酬を予測する報酬期待度というチーム編成を行うためのパラメータを導入する．エージェントはこれらのパラメータを学習適応させながらチーム編成の判断を行う．学習の進行により，エージェントはタスクに対する適切なチームが編成でき，システム全体として処理できるタスク数を向上させ，同時にチーム編成に失敗し廃棄されるタスク数を減らせることを示す．

本論文の構成は以下の通りである．次節で関連研究について述べ，第 3 節で提案するチーム編成のモデルを説明する．第 4 節で各種パラメータを用いたチーム編成と報酬配分を行う方法と，報酬に基づくパラメータの学習手法について提案する．第 5 節では，ランダムにチームを編成する手法と本学習を行う手法を同一条件下でシミュレーション実験を行い，単位時間あたりに処理できたタスクの量を比較し，提案する学習手法の有効性を示す．

連絡先: 浜田　大，早稲田大学大学院基幹理工学研究科情報理工学専攻，d.hamada@isl.cs.waseda.ac.jp

2. 関連研究

上記で述べた [2] の他にもチーム編成に関する研究は数多くある．例えば，エージェントに対するタスクや報酬の配分を最適化するという観点からチーム編成を行う研究がある [3] [4]．[3] では，エージェントに対するシステム全体から見たタスクの最適な配分を実現するアルゴリズムを提案している．しかし指数時間のアルゴリズムであるため，現実のシステムに適用することは難しい．本研究はエージェントの数に比例した多項式時間でチーム編成を実現できるため，エージェントの数が多くても対応できる．[4] では，複数のエージェントがチーム編成を行い，タスクを処理する際に得られる報酬の量を最大化する問題について，パレート最適な解を多項式時間で求めている．しかしパレート最適でも，システム全体の報酬を最大化できるとは限らない．

またタスクを処理する際に必要なチームのメンバを交渉により求めるアプローチもある．たとえば [5] では，交渉のヒューリスティクスを導入し，短い時間で最適解に近い答えを出している．しかし [5] のモデルでは一つのタスクにつき複数のサブタスクが存在し，各エージェントはそれぞれのサブタスクに関して処理できるか否かが 2 値のみで表現されており，本研究よ

り単純なモデルになっている．

一方，[6] では，エージェントが階層的な組織構造を持つ場合において，単位時間あたりの効用を増加させるためエージェントに強化学習を行わせて，エージェント同士のリンクを自動生成する．これにより，特定のエージェントへのタスクの集中を防ぎ，タスク処理の効率化を実現している．一方，これまで述べた研究と異なり，本研究では単に報酬だけでなく，自らが他のエージェントに与える報酬配分率やチーム編成の成否率をまとめて学習するアプローチをとっている．これにより，複雑な状況においても対応可能である．また階層的組織構造にも限定していない．

3. 提案モデル

3.1 エージェントとタスク

システム内に存在するエージェントの集合を $A = \{a_1, a_2, ..., a_n\}$ とおく．各エージェント a_i は自身の処理能力に相当するリソースを持ち，$R_{a_i} = \{r^1_{a_i}, r^2_{a_i}, ..., r^p_{a_i}\}$ と表す．システムに要求されるタスクの種類の集合を $T_{base} = \{t_1, t_2, ..., t_m\}$ とおき，タスク t_i の処理に必要なリソースを $R_{t_i} = \{r^1_{t_i}, r^2_{t_i}, ..., r^p_{t_i}\}$ とおく．タスク t_i を処理することでエージェントに与えられる報酬を u_{t_i} とおく．報酬 u_{t_i} はタスクが必要とするリソース量の合計 $u_{t_i} = \sum_{k=1}^{p} r^k_t$ と考える．

エージェントが処理すべきタスクの集合を $B = \{t_1, t_2, ..., t_b\}$（ただし，$t_i \in T_{base}$）と表す．また，タスク集合を処理するためのエージェントの集合 $G \subseteq A$ に対し，各種リソースにおいてチーム内のエージェントのリソース量の合計が各タスクの必要リソース量の合計を上回ったとき，つまり，式

$$\sum_{a \in G} r^k_a \geq \sum_{t \in B} r^k_t, \forall k, 1 \leq k \leq p \tag{1}$$

が成り立つときタスクの処理が可能とする．このとき G はタスク集合 B のチームと呼ぶ．

システム内における時間の最小単位をターンと呼ぶ．1 ターン中では，タスクがシステムに追加された後，システム内のエージェント全てが並列に行動を行う．このとき，各タスク t_i は確率 p_{t_i} でシステムに追加されるものとする．システムのタスクはキューとして管理され，各エージェントはキューの先頭から処理するタスクを取得する．

3.2 チーム編成

本モデルにおけるチーム編成では，あるエージェントがシステム内の他のエージェントにチーム編成提案を送り，それに対する応答によってチームのメンバを決める．ここで，あるチーム編成の過程において，提案を行ったエージェントをリーダ，提案を受け取ったエージェントをメンバと呼ぶ．

リーダは処理するタスクをキューの先頭からいくつか選択し，そのタスク集合を処理するエージェントを選択する．次に，選択したエージェント全員に提案を行い，メンバからの応答を待つ．そして，リーダからの提案を受諾するエージェントとリーダ自身が合わさりチームとなる．この状態で選択したタスクの集合をチームが処理できれば（すなわち (1) 式が成り立てば）チーム成立となり，チーム編成が成功したことをチームのエージェント全員に知らせる．その後，タスク集合を実行し，チームを解散する．このときタスク集合に含まれるタスクが実行されたと考え，そのタスクの数をタスク処理数と呼ぶ．もし，(1) 式が成立しなければチーム不成立となり，チーム編成の失敗をチームのエージェント全員に知らせ，タスク集合を破棄し，チームを解散する．このとき廃棄されたタスクの数を廃棄タスク数と呼ぶ．チーム編成では，タスク処理数が多く，廃棄タスク数が少ないほど好ましい．

3.3 エージェントの状態

本モデルにおいてエージェントはアイドル状態，提案仮受託状態，提案要求状態，タスク実行状態の 4 つの状態を持つ．エージェントの初期の状態はアイドル状態である．エージェントは状態に応じた行動を取り，次の状態へと遷移する．以下，各状態におけるエージェントの行動について述べる．

アイドル状態では，まずエージェントは自身の能力に見合った量のタスクをキューから取得する．現在システムのキューにあるタスクを先頭から順番に $T_{now} = \{t_1, t_2, ..., t_n, ...\}$ とおくと，エージェント a_i が先頭から取るタスクは $B = \{t_1, t_2, ..., t_{take}\}$ とする．ただし，$take$ は次の条件を満たす n の最大値で決定する．

$$\sum_{k=1}^{n} \sum_{j=1}^{p} r^j_{t_k} \leq \sum_{j=1}^{p} r^j_{a_i} \times 2 \tag{2}$$

次に，エージェントはこのタスク集合を処理できるエージェントの集合を選び，それを仮チームとする．その上で，自身がリーダとなるか，他のエージェントからのチーム編成の提案のうちの一つを受託するかを判断する．この判断の方法に関しては次節で述べる．このとき，受けた提案が一つもない場合は，自動的にリーダとなる．リーダとなる場合は，仮チームのメンバ全員に提案を行い，提案要求状態へ遷移する．リーダとならずに提案を受託する場合は，選択した提案を受託する旨を知らせるとともに，それ以外の提案については提案を拒否することを知らせ，提案仮受託状態へ遷移する．提案仮受託状態とはリーダからの提案を受託したエージェントが，実際にチームが成立するかどうか，リーダからの返事を待っている状態である．チームのリーダからチーム編成成功の知らせを受けると，実際にタスクを実行するためにタスク実行状態となり，チーム編成失敗が知らされたときは，アイドル状態へ遷移する．

提案要求状態とは，リーダが提案を行い，各メンバからその返答を待っている状態である．提案を受託する旨の応答があったエージェント全てと自分自身を合わせチームとする．選んだタスク集合をチームが処理できる場合はチーム成立となり，メンバ全員にチーム編成成功を知らせ，タスク実行状態へ遷移する．そうでない場合はメンバ全員にチーム編成失敗を知らせ，アイドル状態へ遷移する．

タスク実行状態は，タスクを実行している状態であり，一定時間の経過後，アイドル状態へ遷移する．

4. エージェントの行動選択と学習

エージェントはアイドル状態において処理するタスクとチーム，受託するチーム編成提案を選択し，リーダとなった場合はタスク実行状態においてチームのエージェントに対する報酬配分を選択する．エージェントは可能な行動を評価し，その評価値に基づいて次の行動を決定する．行動の評価のため，本研究ではエージェントに対して欲張り度，提案受託期待度，報酬期待度というパラメータを導入する．

4.1 欲張り度

欲張り度とは，エージェントがリーダとなりチームとして得られた報酬のうち，リーダ自身の取り分とする割合で，0 から 1 の値をとる．エージェント a の欲張り度を $Greedy_a$ と書く．チームのリーダとなったエージェント a がタスクを実行したと

き，報酬 $u = u_{all} \times Greedy_a$ を得る．ただし，u_{all} は処理したタスク集合に対する報酬である．また，チームのリーダ以外のエージェントは報酬 $u = u_{all} \times (1 - Greedy_a)/memberSize$ を得る．ただし，$memberSize$ はチームのリーダを除いたエージェントの数である．

リーダとしての取り分を学習するために，欲張り度はチーム編成の成否によって上下させる．チームのリーダとなったエージェント a の欲張り度 $Greedy_a$ は提案要求状態においてチーム編成の成否が決まったとき，以下のように更新する．

$$Greedy_a = (\alpha_g \times S) + (1 - \alpha_g) \times Greedy_a \tag{3}$$

ただし，α_g は欲張り度の学習率（$0 \leq \alpha_g \leq 1$），S はチーム編成が成功した場合は 1，チーム編成が失敗した場合は 0 の 2 値を取る．

4.2 報酬期待度

報酬期待度とは，リーダとなるエージェントからのチーム編成提案を受諾した際の報酬の量の期待度である．この値は全てのエージェントが他のエージェントに対して持ち，エージェント a のエージェント a_{obj} に対する報酬期待度を $RAR_a^{a_{obj}}$ と書く．アイドル状態においてエージェントが受託する提案を選ぶ際，提案メッセージ M ごとに以下の値を計算し，その値によるルーレット選択を用いて受託するチーム編成提案を決定する．

$$U_m^M = \sum_{t \in M_B} u_t \times RAR_a^{sender_M} \tag{4}$$

ただし，M_B はチーム提案におけるタスク集合，$sender_M$ はリーダとなるエージェントである．

報酬期待度はタスクの実行後にチームのリーダから配分される報酬により更新する．エージェント a がリーダ a_l から報酬 $gainedUtility$ を受け取ったとき，報酬期待度 $RAR_a^{a_l}$ は以下のように更新する．ただし，リーダ a_l がチーム編成に失敗したときの報酬 $gainedUtility$ は 0 とする．

$$RAR_a^{a_l} = (\alpha_{RAR} \times gainedUtility) + (1 - \alpha_{RAR}) \times RAR_a^{a_l} \tag{5}$$

ただし，α_{RAR} は報酬期待度の学習率（$0 \leq \alpha_{RAR} \leq 1$）である．

アイドル状態においてリーダとなるかメンバとなるかを判断するために，選択したタスク集合 B の報酬の期待値とチーム編成提案 M を受理したときの報酬期待値を比較する．具体的には以下の通り 2 つについて，欲張り度と報酬期待度を用いて計算する．

$$U_l = \sum_{t \in B} u_t \times Greedy_a \tag{6}$$

$$U_m = \sum_{t \in M_B} u_t \times RAR_a^{sender_M} \tag{7}$$

もし，$U_l \geq U_m$ ならば自身はリーダとなり，$U_l < U_m$ ならば提案されたチームを 1 つ選択し受託する．

4.3 提案受託期待度

提案受託期待度とは，エージェントごとの自身の提案の受諾されやすさを示す度合いであり，この値が大きいほど，そのエージェントに対するチーム編成提案が受理されやすいことを示す．この値は全てのエージェントが他のエージェントに対して持つ．あるエージェント a のエージェント a_{obj} に対する提案受託期待度を $EPA_a^{a_{obj}}$ と書く．エージェント a はタスク集合を処理できる仮チームを選ぶ際，いつくつかの仮チームの候補ごとに以下の評価値を計算し，その値によるルーレット選択を用いて仮チームを決定する．ここで B は提案するタスクの集合，G は仮チームとなるエージェントの集合である．

$$\prod_{j=1}^{p} \left(\sum_{a_m \in G} (r_{a_m}^j \times EPA_a^{a_m}) \div \sum_{t \in B} r_t^j \right) \tag{8}$$

提案受託期待度はチーム編成提案に対する各エージェントの応答によって以下のように更新する．

$$EPA_a^{a_m} = (\alpha_{EPA} \times S_a) + (1 - \alpha_{EPA}) \times EPA_a^{a_m} \tag{9}$$

ただし，α_{EPA} は提案受託期待度の学習率（$0 \leq \alpha_{EPA} \leq 1$），$S_a$ はエージェント a が提案を受託した場合は 1，拒否した場合は 0 とする．

5. 実験と考察

5.1 実験

本研究では，パラメータを更新しない手法（以降，ランダム選択手法と呼ぶ），本研究では前節で述べた欲張り度，提案受託期待度，報酬期待度を学習する手法（以降，提案手法と呼ぶ）を比較した．それぞれの手法について，以下の表に示すパラメータを用いて実験を行った．

表 1: 実験におけるパラメータの一覧

パラメータ	値
エージェントの数	10
タスクの種類数	10
リソースの種類数	2
エージェントの持つ各リソース量	1 〜 10 のランダム
タスクの要求する各リソース量	1 〜 10 のランダム
各タスクの発生確率	0.5
欲張り度の初期値	0.5
提案受託期待度の初期値	0.5
報酬期待度の初期値	0.5
試行ターン数	10000
α_g	0.05
α_{EPA}	0.1
α_{RAR}	0.1

本実験では，50 ターンごとに，タスク処理数，廃棄タスク数をそれぞれ記録した．この実験を 10 回繰り返し，各 50 ターンごとに平均を取ったものを結果とした．

図 1 に 50 ターンごとのタスク処理数の推移，図 2 に 50 ターンごとの廃棄タスク数の推移を示す．図 1 より，提案手法の方がランダム選択手法より，単位時間あたりにおよそ 60%ほど多くタスクを処理できることが分かる．また，図 2 においては，提案手法の廃棄タスク数はランダム選択手法の半分以下程度に抑えられている．

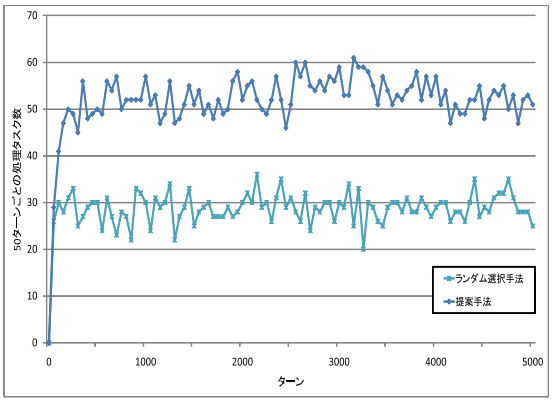

図 1: 50 ターンごとのタスク処理数の推移

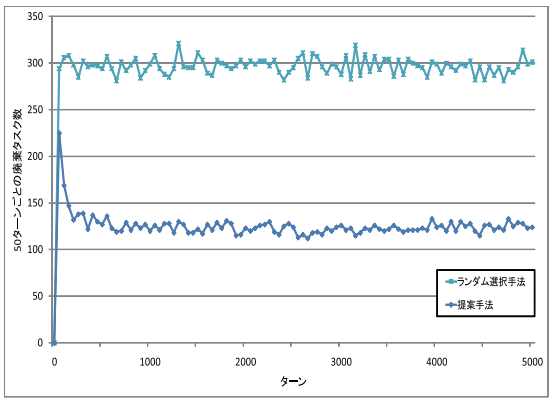

図 2: 50 ターンごとの廃棄タスク数の推移

5.2 考察

本モデルにおいて，リソースを多く持つエージェントはリーダとなる確率が高くなり，その結果他のエージェントにチーム編成提案を送る．一方，エージェントがメンバとしてチーム編成提案を選ぶ際は，タスク集合の報酬が多いものを高確率で選択する．本モデルにおいてタスクの報酬と必要リソース量は比例するため，結果的に最もタスクの必要リソース量の多いチーム編成提案，すなわち最も多くのリソースを持つエージェントからの提案が高い確率で受託される．ランダム選択手法ではパラメータを学習しないためこの構造が維持される．しかし，本モデルにおいてこの戦略は最適ではない．なぜなら，エージェントが一度に取得できるタスクの量には制限があり，リーダとなるエージェントの数が少ないとリソースが余るからである．

一方，提案手法は提案受託期待度と報酬期待度の学習によりそれぞれチーム編成提案を送るエージェント，提案を受けるエージェントが固定化し，チーム編成が効率的に行われるようになる．その結果，あるリーダからチーム編成提案を受け取らなくなったエージェントは別のチームが編成できるようになる．このようにシステム内に複数のチームができることにより，リソースを分配してタスク処理が行えるようになったと考えられる．

なお本研究では，他のエージェントのリソースは既知で，その情報を元に仮チームを作っていたが，システム内の他のエージェントの種類や能力を未知とする研究もある．例えば [7] では，他のエージェントのタイプを未知として，強化学習を行わせつつ繰り返し協調を行うことにより，システム内の他のエージェントのタイプを学習する手法を提案している．本研究ではシステム内のエージェントの能力を既知としているが，[7] の手法を用いることにより，エージェントの能力が未知の場合においても本研究で対応が可能であると考えている．

6. 結論

本研究ではチーム編成問題において，エージェントに報酬の配分を決める欲張り度，提案したチームへの参加受理の度合いを示す提案受託期待度，チームへ参加したときに受ける報酬を予測する報酬期待度というパラメータを導入し，それを学習させ，チーム編成を効率化する手法を提案した．また使用したモデルは，エージェントが互いにメッセージを送りあうことによりチーム編成を実現した [2] で提案されたものを，発生したタスクを公平に選択するように改良したものである．その上で評価実験を行った結果，廃棄タスク数を減らしつつタスク処理数が上昇することを確認した．今後は，より詳細な実験を行うとともに，[6] のようにネットワーク構造（ただし階層構造のみではなく）を取り入れたモデルや [7] のようにエージェントのリソースを推定しながら学習する手法などを検討したい．

参考文献

[1] 総務省. 平成 20 年版 情報通信白書.

[2] Thomas Genin. “Coalition Formation Strategies for Self-Interested Agents in Task Oriented Domains.” *Proc. of IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, 2010.

[3] O. Shehory and S. Kraus. “Methods for task allocation via agent coalition formation.” *Artificial Intelligence*, 101(1-2):165-200, 1998.

[4] O. Shehory and S. Kraus. “Feasible Formation of Coalitions among Autonomous Agents in Nonsuperadditive Environments.” *Computational Intelligence*, 15:218-251, 1999.

[5] S. Kraus, O. Shehory, and G. Taase. “Coalition formation with uncertain heterogeneous information.” *Proc. of AAMAS ’03*, July 2003.

[6] 片柳 亮太, 菅原 俊治. マルチエージェント環境におけるチーム編成の効率化へ組織構造が与える影響の分析. JAWS2009 予稿集, 2009.

[7] G.Chalkiadakis and C.Boutilier. “Sequential decision making in repeated coalition formation under uncertainty.” *Proc. of AAMAS ’08*, 347-354, 2008.