RF-010

報酬に基づく強化学習を用いたチーム編成手法の提案と評価

Utility-based coalition formation methods using reinforcement learning and their evaluation

浜田 大†
Dai Hamada

菅原 俊治†
Toshiharu Sugawara

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．一般にネットワーク上のサービスは複数のサービス要素により実現される．したがってサービスを実現するタスクは複数のサブタスクから構成されると考えられ，これらサブタスクを全て処理することで，タスクの処理が完了する．たとえ 1 つのサブタスクの処理が失敗あるいは遅れ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．

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，[2] 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いる一方で，エージェントのスキルや報酬の値によっては一回も処理されないタスクが生じる可能性があるため，現実とは合致しないケースもある．また手法の性能指標として，チーム編成の成功率を確かめ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単位時間あたりのタスク処理量については議論されていない．

本研究では [2] のモデルを改良し，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し，タスクを処理した際に得られる報酬をエージェントで一定にする．また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それらの値を元にチームのメンバとなるエージェントを選択したり，チームへの参加を決定する．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て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，提案する学習手法の有効性を示し，各学習による特徴を解析するため，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する．

†早稲田大学理工学術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り，特に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存在する [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．一方，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．しかしシステム全体の報酬を最大化できているとは限らない．

また [5] では，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ている．特に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，本研究と同様に一つのタスクが複数のサブタスクで構成される形になっているが，各エージェントは各サブタスクに関して処理できるか否かが 2 値で表現されており，本研究より単純なモデルとなっている．

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とにより，タスク処理の効率化を行っている．これにより，特定のエージェントへのタスクの集中を防いでいる．一方，本研究は [6] の研究と異なり，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，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 提案モデルと課題

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，$H_{a_i} = \{h^1_{a_i}, h^2_{a_i}, ..., h^p_{a_i}\}$ 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 $N_{t_i} = \{n^1_{t_i}, n^2_{t_i}, ..., n^p_{t_i}\}$ 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} はタスクの必要とするリソース量の合計，つまり $u_{t_i} = \sum_{k=1}^{p} n^k_t$ とする．

エージェントが処理すべきタスクの集合を $B \subseteq T_{base}$ 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，タスクの処理が可能とする．つまり，式

$$\sum_{a \in G} h^k_a \geq \sum_{t \in B} n^k_t, \forall k, 1 \leq k \leq p \quad (1)$$

が成り立つ必要がある．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．したがって，[2] と異なりエージェントは選り好みをせず，要求された順に処理を行う．

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メッセージ（以下，提案メッセージと呼ぶ）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，提案メッセージを送ったエージェントをリーダ，リーダの送った提案メッセージ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．このとき選択されたエージェントをリーダを含めて仮チームと呼ぶ．次に，そのエージェント全員に提案メッセージを送信し，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旨のメッセージが返ってきた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旨のメッセージ（以下，チーム編成成功メッセージと呼ぶ）を仮チームのエージェント全員に送る．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，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表すメッセージ（以下，チーム編成失敗メッセージと呼ぶ）を仮チームのエージェント全員に送った後，タスク集合を破棄し，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廃棄され，その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，提案受託状態，提案送信状態，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．状態遷移の様子を図 1 に示す．4 つの円はエージェントの状態を表し，各エージェントの状態は 1 ターンに一度だけ矢印の方向へ遷移する．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．

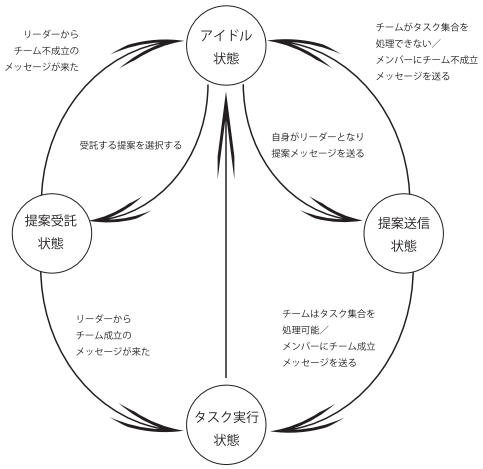

図 1: エージェントの状態遷移

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．

$$\sum_{k=1}^{n} \sum_{j=1}^{p} n^j_{t_k} \leq \sum_{j=1}^{p} h^j_{a_i} \times L \quad (2)$$

ただし，L は正数であり，単独で処理可能なタスク量に対し，チームを組んで行う上限とする，なお，$L = 1$ であればチームを組む必要はない．

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提案メッセージのうちの一つを受託するか選択する．この選択に関しては次節で述べる．このとき，自身に提案メッセージが一つも来ていなかった場合は，提案メッセージの選択は行われず自動的にリーダとなる．もし，リーダとなる場合は，仮チームのメンバ全員に提案メッセージを送り，提案送信状態へ遷移する．リーダとならずに提案メッセージを受託する場合は，そのメッセージを送ったエージェントに対して提案を受託する旨のメッセージを，それ以外の提案メッセージを送ったエージェントに対して提案を拒否する旨のメッセージを送り，提案受託状態へ遷移する．提案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メッセージが来た場合は，実際にタスクを実行するためにタスク実行状態へ，チーム編成失敗メッセージが来た場合は，アイドル状態へ遷移する．

提案送信状態とは，リーダとなり提案を送信したエージェントが，各メンバから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．チームがアイドル状態において選んだタスク集合を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メッセージを送り，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メッセージを送り，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させるために，欲張り度はチーム編成の成否によって上下する．チームのリーダとなったエージェント a の欲張り度 $Greedy_a$ は提案送信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．

$$Greedy_a = (\alpha_g * SUCCESS) + (1 - \alpha_g) * Greedy_a \quad (3)$$

ただし，α_g は欲張り度の学習率 $(0 \leq \alpha_g \leq 1)$，$SUCCESS$ 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，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つ．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．

$$U_{member}^{M} = \sum_{t \in M_B} u_t \times RAR_a^{sender_M} \quad (4)$$

ただし，M_B は提案メッセージ M におけるタスク集合，$sender_M$ は提案メッセージを送信したエージェントである．

報酬期待度を学習するため，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させる．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．

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

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．

$$U_{leader} = \sum_{t \in B} u_t \times Greedy_a \quad (6)$$

$$U_{member} = \sum_{t \in M_B} u_t \times RAR_a^{sender_M} \quad (7)$$

もし，$U_{leader} \geq U_{member}$ ならば自身はリーダとなり，仮チームに対して提案メッセージが送られる．$U_{leader} < U_{member}$ ならば選択した提案メッセージを受託する．

4.3　提案受託期待度

提案受託期待度とは，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通りやすいことを示す．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 はアイドル状態においてタスク集合を処理できる仮チームを選ぶ際，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．提案するタスクの集合を B とおくと，仮チームとなるエージェントの集合 G の評価値を以下の式で表す．

$$\prod_{j=1}^{p} \Big(\sum_{a_m \in G} (h_{a_m}^j \times EPA_a^{a_m}) \div \sum_{t \in B} n_t^j \Big) \quad (8)$$

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．

$$EPA_a^{a_m} = (\alpha_{EPA} * SUCCESS_a) + (1 - \alpha_{EPA}) * EPA_a^{a_m} \quad (9)$$

ただし，α_{EPA} は提案受託期待度の学習率（$0 \leq \alpha_{EPA} \leq 1$），$SUCCESS_a$ 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 の 2 値をとる．

4.4 各パラメータ学習の組み合わせ

本研究では前節で述べた 3 つのパラメータ（欲張り度，提案受託期待度，報酬期待度）の中で，更新するパラメータの組み合わせにより 7 通りの手法を用いる．欲張り度，提案受託期待度，報酬期待度を更新することをそれぞれ「G」「E」「R」で表し，アルファベットを並べることで複数のパラメータを学習することを表す．例えば GE 手法とは，欲張り度と提案受託期待度を更新する手法，GRE 手法とは欲張り度，報酬期待度，提案受託期待度を更新する手法，E 手法とは提案受託期待度のみを更新する手法である．

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），G 手法，R 手法，E 手法，GR 手法，GE 手法，RE 手法，GRE 手法を比較した．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

また，各学習パラメータの学習率 α は以下の表のように設定した．なお，学習を行わない場合は，対応する学習率を 0 とする．例えば G 手法では，$\alpha_{EPA} = 0$，$\alpha_{RAR} = 0$ とする．

表 2: 学習パラメータの学習率

学習率	値
α_g	0.05
α_{EPA}	0.1
α_{RAR}	0.1

本実験では，50 ターン実行するごとに，処理できたタスク数（以降，タスク処理数と呼ぶ），チーム編成に失敗し，実行がキャンセルされたタスク数（以降，廃棄タスク数と呼ぶ）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 2 にランダム選択手法，GRE 手法，G 手法，R 手法，E 手法の 50 ターンごとのタスク処理数の推移，図 3 に 50 ターンごとの廃棄タスク数の推移を示す．

図 2 より，GRE 手法，および G 手法におけるタスク処理数は，ランダム選択手法よりも 1.6 倍以上大きい値で収束している．また，GRE 手法は G 手法よりも 1.2 倍ほど高い値となる．これは，欲張り度のみの学習よりも，提案受託期待度と報酬期待度をあわせて学習する方が効果が出ることを示している．また，E 手法に関してはランダム戦略手法とほぼ同じ結果になり，R 手法は 0 に収束した．

図 3 より，G 手法における廃棄タスク数は 110 前後の値で収束しており，本提案手法の中では最も低い値となった．GRE 手法では 120 前後の値となり，ランダム選択手法の半分以上低い値で収束した．R 手法に関しては高い値を示し，500 ほどの値で収束した．E 手法の廃棄タスク数はランダム選択手法とほぼ同じ結果になった．

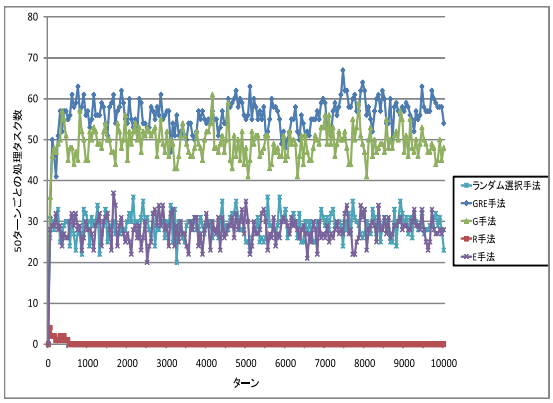

図 2: ランダム選択手法，G 手法，R 手法，E 手法における 50 ターンごとのタスク処理数の推移

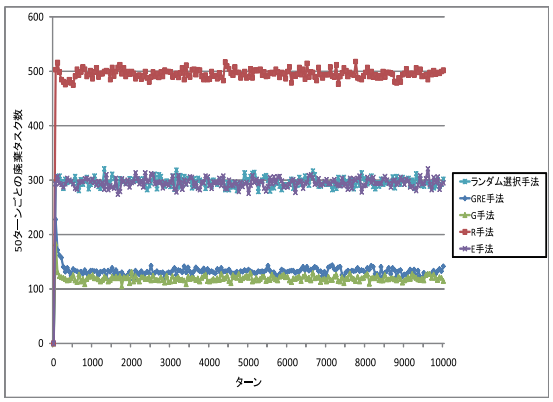

図 3: 50 ターンごとの廃棄タスク数の推移

図 4 にランダム選択手法，GRE 手法，GR 手法，GE 手法，RE 手法の 50 ターンごとのタスク処理数の推移，図 5 に

50 ターンごとの廃棄タスク数の推移を示す．図 4 より，GR 手法と GRE 手法における 50 ターンごとのタスク処理数は 55〜60 とほぼ同じ結果になった．GE 手法は GRE 手法よりもやや小さい値を示した．RE 手法に関しては低い値で収束した．図 5 より，GRE 手法と GR 手法，GE 手法における 50 ターンごとの廃棄タスク数は共に 120 前後の値となった．RE 手法はランダム選択手法よりも高い値を示し，480 ほどで収束した．

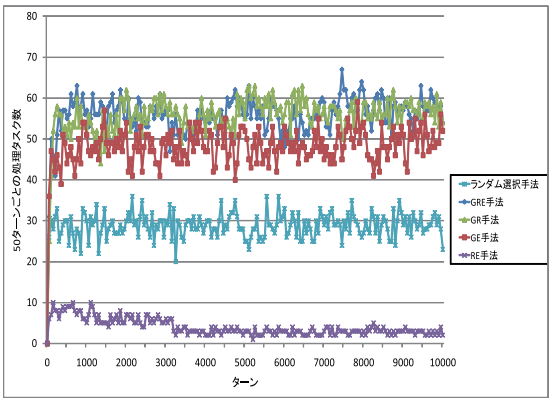

図 4: 50 ターンごとのタスク処理数の推移

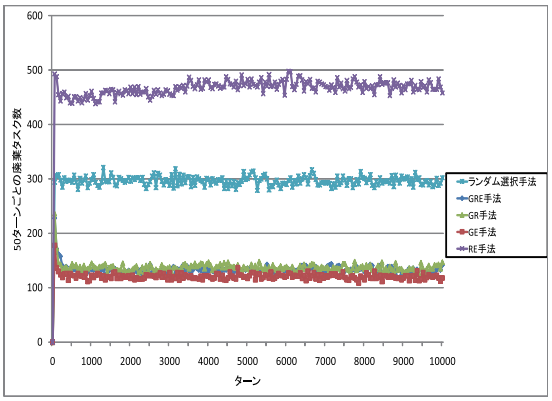

図 5: 50 ターンごとの廃棄タスク数の推移

5.2 考察

実験より，GRE 手法の単位時間あたりのタスク処理数が最も多いことが分かった．また，最も単位時間あたりの廃棄タスク数が少ないものは G 手法であった．G 手法では，初めのうち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．欲張り度を学習する場合は，リソースの少ないエージェントがリーダとなりチーム編成提案を送った場合，チーム編成に失敗しやすく，欲張り度が下がるため，以降はリソースを多く持つエージェントからの提案を受諾しやすくなる．このように，リソースを多く持つエージェントがリーダとなり全員にチーム編成提案を送る戦略が維持される．その結果，チーム編成が安定化し，ランダム選択手法よりも良い結果が得られた．しかし，本モデルにおいてこのような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．

R 手法では G 手法と同じく，リソースを多く持つエージェントはリーダとなりやすく，他のエージェントに提案メッセージを送り，他方各エージェントもその提案を受理しやすい．しかし，チームの規模が大きくなると，メンバとなるエージェント一体あたりに配分される報酬は小さくなる．したがってメンバとなっていたエージェントの報酬期待度は下がり，一部のエージェントは自身がリーダとなることを選択する．同時にリーダとなるエージェントも多くのエージェントに提案メッセージを送る確率が高くなり，その提案を受理したエージェント一体あたりに配分される報酬は小さくなる．また，提案が受理されなくなったリソースを多く持つエージェントは欲張り度を学習しないため，自身よりリソースの少ないエージェントの提案を受理することはせず，引き続きリーダとなり，全てのエージェントに提案メッセージを送り続ける．このようなプロセスが繰り返され，リーダとなるエージェントの数が増加したため，処理タスク数が減少したと考えられる．

E 手法では結果がランダム選択手法と変わらなかった．これは，エージェントが欲張り度および報酬期待度を学習しないため，ランダム選択手法におけるチーム編成の成否に影響しなかったと思われる．

また，GE 手法は G 手法に加えて提案受託期待度を学習したものであるが，結果はタスク処理数，廃棄タスク数共に G 手法と変わらなかった．これは，エージェントが報酬期待度を学習しないためであり，リーダから配分される報酬が少なくても，そのエージェントからの提案を受託し続けるからである．

GR 手法は G 手法に加えて報酬期待度を学習したものであり，タスク処理数の結果は G 手法よりも向上した．これは次のようなメカニズムが働いたためと思われる．まず，G 手法と同じプロセスを経てリーダから配分される報酬が少なくなった場合，エージェントは報酬期待度を学習し，そのリーダからの提案を受託しにくくなる．そのエージェントはリーダとなる確率が高まるが，リソースを多く持つ場合はチーム編成の成功率は高くなり，徐々に自身の欲張り度が上昇し，メンバに分配する報酬が減少する．リソースが少ない場合はチーム編成提案が失敗しやすく，自身の欲張り度が減少するため，他のエージェントからの提案を受託するようになる．このようにして，リソースを多く持つ複数のエージェントが交代でリーダとなる現象が繰り返されたと考えられる．

最後に，GRE 手法は GR 手法に加え，提案受託期待度を学習したものである．リーダから配分される報酬が少なくなりリーダとなるエージェント a に関しては，その間に他の

エージェントから送られたチーム編成提案を拒否するため，エージェントからチーム編成提案が送られる確率が減少する．したがって，a はそのままリーダとなり続け，提案受託期待度の学習により適切なチームを編成できるようになる．このとき，a の持つリソースが少なくてもリーダとなり続けられる確率が高い．このように，エージェント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，複数の計算機による協調負荷分散をマルチエージェントシステムでシミュレートした．協調手法としては，各計算機をエージェントに見立て，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] の手法を用いたものの，発生したタスクを公平に選択するモデルとした．またタスクを処理した際に得られる報酬をエージェントで一定にした．さらに，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チーム編成を行うためのパラメータを導入し，それを学習させ，マルチエージェントシステム全体としての効率を向上させることを提案した．

そして改良したモデルを元に，欲張り度，提案受託期待度，報酬期待度を学習する手法を提案し，シミュレーション実験を行った．その結果，廃棄タスク数を減らしながらタスク処理数が挙がることを確認した．また，タスク処理数の改善には欲張り度，提案受託期待度，報酬期待度のパラメータを全て学習することで，最大の効果が得られた．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．

なお、本研究の一部は科研費 (22300056) の助成を受けている．

参考文献

[1] 総務省. 平成 20 年版 情報通信白書.

[2] Thomas Genin, Samir Aknine, “Coalition Formation Strategies for Self-Interested Agents in Task Oriented Domains,” wi-iat, , *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, vol. 2, pp.205-212, 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. 強化学習とリンクの動的生成を用いた組織の再構成によるチーム編成の効率化. 2009 年度人工知能学会全国大会, 2G2-2, 松山, Jun. 17-19, 2009.

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