マイクロタスク型クラウドソーシングにおける ワーカの希望タスク数を考慮したタスク割当て手法

山口　大河†　　鈴木　伸崇††

† 筑波大学情報学群知識情報・図書館学類　〒 305–8550 茨城県つくば市春日 1-2
†† 筑波大学図書館情報メディア系　〒 305–8550 茨城県つくば市春日 1-2
E-mail: †s1711579@n-univ.ac.jp, ††nsuzuki@slis.tsukuba.ac.jp

あらまし　マイクロタスク型クラウドソーシングでは、不特定多数の参加者（ワーカ）を効率よくタスクに割り当てる手法が必要となる。そのため、ワーカ割り当て手法に関する研究は多く行われているが、その多くは生産性やスループットなどのリクエスタの事情を優先したものである。しかし、場合によっては成果を上げることよりも、より多くの人が参加し寄与する、参加したという満足感をワーカに与えるという点が重視されることもある。例えば、ボランティアベースのクラウドソーシングがそれにあたる。このような場面の割当て手法としてインクルージョン性を優先した割当て手法が提案されている。本研究では，インクルージョン性の意義を損なうことなく，仕事量の観点でよりワーカ側の事情に配慮した新たな割当て手法を提案する．

キーワード　マイクロタスク型クラウドソーシング，タスク割当て，深さ制限探索

1　はじめに

近年, 新たな目的遂行の手段としてクラウドソーシングが注目を集めている. クラウドソーシングとは, 不特定多数の寄与のもと目的を遂行するプロセスのことを指し, 目的に応じた形をとることが可能という特徴がある. その形式は, デザイン案などに用いられるコンペティション型,web 開発などに用いられるプロジェクト型, データ収集などに用いられるマイクロタスク型に大別される．本研究で取り扱うのはマイクロタスク型であり, これは単純な作業にタスクを分割しそれを複数のワーカで分担して行うことで目的を遂行する形式をとる. マイクロタスク型はワーカにタスクを分担させるという性質上, 募集したワーカにタスクを割り振るという行程が必要になることがある. そのような行程をタスク割当てと呼び, 近年多くの研究が行われている．

そうした研究の多くは, 良いタスク割当てを, 結果のデータ品質が良いもの, 目的遂行に必要なコストが低く抑えられるもの, 必要となる時間が少なく済むもの, などのリクエスタ側の事情に則した観点から定義している．一方で, 近年ワーカ側の事情に則した観点から良いタスク割当てを定義している研究が注目されている．そのうちの一つとして，できるだけ多くのワーカに同じようにタスクが分担されるもの, を良いタスク割当てと定義する研究 [1] が存在し, このような評価指標を「インクルージョン性」と呼んでいる. しかし, インクルージョン性は前述のリクエスタ側の指標とトレードオフの関係にあることが知られている．

このインクルージョン性に焦点を当てたタスク割当て手法の研究 [1] では, 生産性やスループットを著しく下げることなく, インクルージョン性を高める割当て手法が提案された. この研究において, インクルージョン性は各ワーカに割り当てられたタスクインスタンスの分散と定義された. また, 同研究において提案された手法ではワーカの有する能力とタスクに要する能力を考慮した割当てを可能にしている．インクルージョン性を考慮したタスク割当て手法は以下のような場面において有用である．まず，ワーカ側の事情を考慮した割当て手法はボランティアベースの場面において重要である．次に, インクルージョン性はソーシャルインクルージョンの文脈とクラウドソーシングを結びつけるうえで重要な視点である．

本研究では，インクルージョン性の「できるだけ多くのワーカに同じように仕事を分担する」 [1] という定義の解釈を拡張する. 同研究において,「同じように仕事を分担する」とは同じ程度の量の仕事を分担すると解釈されている. 本稿では, これを「希望に沿った量の仕事を分担する（同じ程度の希望であれば同じ程度の仕事量を分担する）」と拡張し, インクルージョン性の意義を損なうことなく, よりワーカ側の事情を反映することが可能なタスク割当て手法を提案する. これにより,「忙しさ」などの個々の事情,「暇つぶし」「金銭目的」などのモチベーションの違いといった潜在的な要素を反映したうえで, ワーカに配慮した割当てを可能とする. 以降本稿では, 区別のため, 前述の拡張されたワーカへの配慮を「Consideration」, ワーカに漏れなくタスクが割当てられていることを「Inclusion」と呼称する. また一方で, このような割当てを行う場合, ワーカのスキルによらずタスクを希望通りに割り当てる必要があり, 生産性やスループットを低下させる可能性があるが, この点に関しては今後の課題とし, 本研究では Inclusion を損なうことなく,Consideration を高めることを可能とする割当て手法の提案を目的とする．

本研究ではまず，ワーカの希望を「希望に沿った量のタスクを分担すること」と想定し，ワーカが 1 から 5 の 5 段階の希望度を指定できるというモデルを作成した．そのモデルにおいて,

希望度に応じてタスクを割り当てる問題の計算の複雑さについて考察した．その結果，ワーカの希望度に応じて最適な形でタスクを割り当てることが NP 困難性をもつことを，最大流問題の 1 つである MFCG 問題から帰着することにより証明した．この結果から，最適な割当てを効率よく求めることは困難であることがわかった．そこで本研究では，ワーカに割り当てられたタスク数とワーカの希望タスク数からスコアを計算し，そのスコアに基づいて深さ制限探索を繰り返し行うことで，最適に近い割当てを効率よく行うアルゴリズムを構成した．

評価実験として，複数通りのテストデータを作成し，提案手法を実行し，評価を行った．得られた結果は，Inclusion を損なうことなく，Consideration を高めるという本研究の目的に対して有効であることを示すものであった．

2 モ デ ル

2.1 タスクとタスクインスタンス

本稿では，あるプロジェクトに対して，目的遂行に必要な行程を複数の作業に分割することを考え，その各作業をタスクと呼ぶ．また，このように定義される各タスクに対して，実際にワーカが行う 1 単位の作業をタスクインスタンスと呼ぶ．具体的には,「ある音声データを文字データに起こす」というプロジェクトに対し,「音声データを文に分割する」「分割された文をタイピングする」という分割された作業がタスク,「（音声データ中の）ある 30 秒の音声データを文に分割する」「（分割された）ある 1 文をタイピングする」等のより具体化された作業がタスクインスタンスである．つまり，各タスクには複数のタスクインスタンスが属することになる．また，各ワーカにはタスクインスタンス単位で作業が割当てられ，同タスクに複数のワーカが割当てられることはあるが，同タスクインスタンスに複数のワーカが割当てられることはない．本研究では，その作業内容に関わらず，1 単位のタスクインスタンスは同程度の作業量を有するものであると想定する．

2.2 取り扱うワークフローとワーカの設定

本研究で扱うワークフローは，あらかじめ分割されるタスクが決定されているものを考慮する．つまり，募集されたワーカ群とその有する能力に応じて有機的にワークフローを作り替えるようなことは行わない．ただし，ワークフローは，同一ではない複数の実行可能な経路から構成されるものとする．ここで同一な経路とは，必要となる能力の組が一致するようなタスクを同一のタスクとみなし，同一なタスクのみから構成される経路のことをいう．

ワーカは，プロジェクトに対して募集が行われ，募集の際にはワークフロー内で必要になる能力について，各能力の有無を把握できているものとする．この際，ワークフロー内に割当て可能なタスクが 1 つ以上存在するワーカのみを受け入れる．また本研究では，ワーカは募集の際にプロジェクトにどの程度寄与することを希望するか（どの程度の量の作業を行いたいか）を 1 から 5 の 5 段階で希望可能であることを想定し，これをワーカの希望度と呼ぶ．数字が大きいほど割当てられるタスクインスタンス数は増加し，希望度は質的でなく量的であると説明が行われているものとする．(希望度が 2 倍，3 倍となると割当てられるタスクインスタンスは 2 倍，3 倍となる)．本研究では以降，希望度は量的な指標として評価・実験を行う．また，本研究ではワーカの増減，希望度の増減は割当て途中では行われないものとする．

2.3 本研究で用いる略式記号

本研究では，説明の都合上，以下のような略式記号を使用する．

- タスクが必要とする・ワーカの有する能力の集合:$Abilities = \{a_1, a_2, a_3, \ldots\}$
- 募集されたワーカの集合:$W = \{w_1, w_2, w_3, \ldots\}$
- ワークフロー:$Workflow = (V, E)$
- ワーカの希望タスク数:$MT(w)\,(w \in W)$
- ワーカのスコアの計算式:$Formula$
- 深さ制限探索の設定深度:$Depth$

本稿では,$Formula$ を以下のように定義する．

$$Formula = MT(w) - allocation\_num(w) + Correction\_value$$

また，$allocation\_num(w)$ はワーカ w に現在割り当てられているタスク数を，$Correction\_value$ は補正値を表す．補正値は Inclusion を損なうことがないように，タスク未割当てのワーカに対して正の値を与える．これにより，タスク未割当てのワーカのスコアを高く計算し，優先的にタスクを割り当てる．

3 評価指標について

本研究では，インクルージョン性（$inclusion$）とワーカ不満度（$dissatisfaction\_rate$）の 2 種類の評価指標を使用する．

本稿で用いるインクルージョン性は本研究の趣旨に合わせ，前述の Inclusion に準拠した定義を使用する．関連研究 [1] では，インクルージョン性は「できるだけ多くのワーカに同じように作業を分担してもらうこと」と定義され，その定義に合わせ，各ワーカに割当てられたタスクインスタンスの分散をインクルージョン性の評価指標として定義した．一方，本研究では全てのワーカに対して同じ量の作業を分担することを目的としていない．そこで，$inclusion$ を各ワーカにタスクが漏れなく割り当てられているかどうかを評価する指標とし 2 て使用し，以下の式で定義する．

$$inclusion = \sum_{w \in W} \frac{(allocation_w - \overline{Allocation})^2}{|W|}$$

ただし，$allocation_w$ はワーカ w に割り当てられたタスクインスタンス数が 0 なら 0，それ以外の値を取るなら 1 が与えられ，$\overline{Allocation}$ はその平均値を表す．この指標は関連研究で用いられる指標をより一般化したものとなっており，この値を 0（タスクを割当てられていないワーカが存在しない）とすることを目標とする．

また本研究では，満足度が高い状況を，ワーカの希望度に応

じた量の作業が該当ワーカに分担されていることと定義する．対して，不満度は希望度から判断される適切な量のタスクインスタンスよりも割当てられたタスクが多い，もしくは不足である状態を評価する指標であり，

$$dissatisfaction\_rate = \sum_{w \in W} \frac{|taskinstances_w - Ideal\_taskinstances_w|}{Ideal\_taskinstances_w \times |W|}$$

と定義する．ここで各ワーカについて，$taskinstances_w$ は実際に割り当てられたタスクインスタンスの量，$Ideal\_taskinstances_w$ は希望度から算出した適切と想定されるタスクインスタンスの量であり，$Ideal\_taskinstances_w = All\_taskinstances \times ratio_w$ で計算したものである．$All\_taskinstances$ は全体で割当てられたタスクインスタンスの総数，$ratio_w$ は全ワーカの希望度の合計値に占めるそのワーカ w の希望度の割合を表している．

4 提 案 手 法

4.1 手法の概要

提案手法はワーカの要望に沿った形でタスクをワーカに割り当てる手法である．本研究で扱うワーカの要望とは前述の希望度のことを指す．本手法では，各ワーカの登録した希望度を疑似的に，希望するタスク数と考え，各ワーカを希望するタスク数の回数だけタスクに割り当てることを目的としている．詳細は次節に記すが，この割当て問題は NP 困難性があり，これを最適な形で効率的に解くことは不可能である．そこで，提案手法では最適解に近い値を効率的に求めることを考える．

4.2 割当ての NP 困難性

本研究で取り上げる割当てを最適な形で行うことは NP 困難性をもつことを示す．本稿では，すでに NP 困難性の証明された MFCG 問題 [3] を利用する．MFCG 問題とはグラフの最大流問題の一つであり，通常のグラフに制約エッジが付与されたグラフにおいて，流量がある値 N 以上であるかどうかを判断する問題である．制約エッジは，通常のグラフにおける異なる二つのエッジ間に対して付与され，制約エッジにより結ばれたエッジは両立しないことを表している．また，MFCG 問題は各エッジの容量が 1 かつ図 1 のような形状に限定しても NP 困難性をもつことが示されている．

定理１：最適な割当てを求める問題は NP 困難である．

証明：以下，各エッジの容量が 1 かつ図 1 の形状をもつグラフ G を利用して証明を進める．まず，グラフ中の各エッジについて，間に新たなノード 1 つを加える操作を行う（エッジ $v_i \to v_j$ であれば，$v_i \to v_i' \to v_j$ となるような操作を行う）．続いて，このようにして作成されたグラフに対して，制約エッジを追加されたノード間をつなぐように結びなおし，これをグラフ G’とする．図 1 のグラフに含まれるすべてのエッジに対して上述の操作を行ったものを図 2 に示す．ここで，各ノードには一つのタスクが対応し，制約エッジで結ばれたノードには一方にのみワーカを割り当てることができると考えると，「作成されたグラフ G’（図 2）が流量 N を満たすか」という問題は，「$N \times (2h+1)$ 人のワーカを割り当てることが可能か」という問題と同一であると考えることができる．また，後者は以下に示すような条件のもと行われるワーカへのタスク割当てと同一であると考えることができる．

- 同一の能力を持つワーカは一人のみである
- 能力に包含関係は存在しない
- 各タスクに対応するワーカが必ず 1 人のみ存在する
- 制約エッジで結ばれたタスクには同一の能力が必要であり、両端のタスク両方にワーカを割り当てることはできない
- ワーカ集合 W は $k \times (2h+1) - |R|$ 人の集合である
- ワーカの希望タスク数はいずれのワーカについても 1 である
- 最適な状態とは不満度が最小となる割当てであるが，この場合はワーカ w の充足率（$satisfaction\_rate_w$）の合計が最大となる割当てと同値である

ただし，R は制約エッジの集合，Ta_w を有効なパス内のワーカ w への割り当て数，Tr_w をワーカ w の希望するタスク数であるとして，$satisfaction\_rate_w = \frac{Ta_w}{Tr_w}$ である．上述の MFCG 問題の変形によるタスク割当ては，本研究で扱うタスク割当てよりも条件の増えたものとなっている．NP 困難性の証明においては，より条件の多い部分問題において NP 困難であるならば，より一般化された問題もまた NP 困難であると証明できる．よって，本研究で扱う割当ては NP 困難である．$(Q.E.D)$

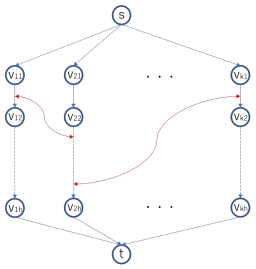

図 1　制限付きグラフ G

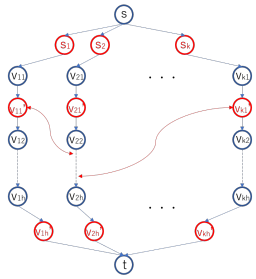

図 2　変換後の制約付きグラフ G'

4.3 深さ制限探索

本節では，深さ制限探索について説明する．深さ制限探索とは，選択可能な複数の事象集合 C に対して，$c \in C$ である事象 c を選択した場合を想定し，同様の手順にて指定回数選択を行った結果から遡り，最適な事象 c’ $(c' \in C)$ を選択する手法である．本研究では，ワーカのスコア化により優先的に割り当てるべきワーカ（以降，高優先度ワーカと呼称する）を算出し優先的に割り当て，これにより生じた複数の割当て候補の中から最適に近いものを選択する目的から，深さ制限探索を採用している．図 3 はその一例である．図 3 は深さ 3 を想定したものであり，各ノードに示された数字は各ノードに至るまでのスコアの合計を表している．図 3 の例では深さ 3 の想定であるため，3 度の分岐が行われ，最も深い「深さ 3 の層」において最適なノードを探索する（この例では「85 のノード」が該当する）．そこで深さ制限探索は，そのようなノードに至る経路を逆算し，「深さ 1 の層」において最適と考えられるノード（図 3

では「19 のノード」）を選択する処理を行う．

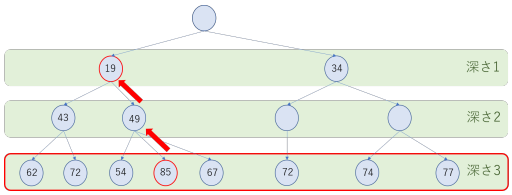

図 3　深さ制限探索例

4.4　割当てアルゴリズム

本節では，まず提案手法の概要を例を用いて説明し，次に提案手法の定義を行う．ワーカ群（図 4）とワークフロー（図 5）に対してアルゴリズムを適用する流れを一例として，図 6 から図 8 に示す．まず，ワーカ群に含まれるワーカに対して現在の割り当て数と希望タスク数からスコアを計算し，スコアの高いワーカから K 人（図中では $K = 4$）のワーカを高優先度ワーカ群として抽出する（図 6）．続いて，ワークフロー内の各パスについて割当て可能な高優先度ワーカの人数を比較し，最大数のワーカを割当て可能なパスを選択する（図 7 ではパス (ア) を選択する）．続いて，選択したパスに対して，高優先度ワーカの割当てられていないタスク（図中のタスク C）に対して割り当て可能なワーカの組み合わせを網羅的に求め，深さ制限探索を行う（図 8）．このような場合，タスク C に対して，W_1，W_2，W_6 を割り当てた場合の 3 通りの割当てを比較し，最適と思われる割り当てを行うことになる．

図 4　ワーカ例

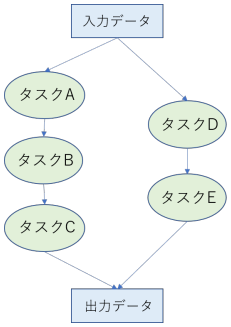

図 5　ワークフロー例

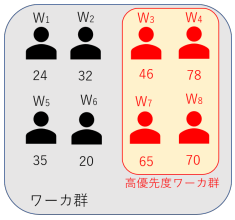

図 6　高優先度ワーカ抽出例

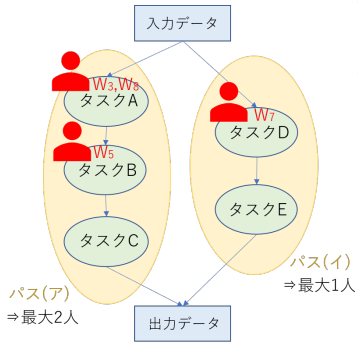

図 7　高優先度ワーカの割当て例

次に，提案手法を示す．まず，提案手法を示すにあたり，アルゴリズム中で使用する関数を定義する．

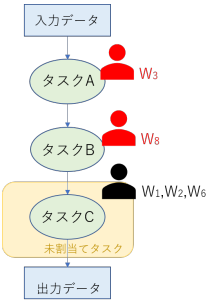

図 8　深さ制限探索によるワーカ割当て例

- *All_Passes(Workflow)*: $Workflow$ 中に含まれる入力ノードから出力ノードに至るまでの網羅的なパスの集合を出力する関数
- *Task_Available(w,t)*: ワーカ w のもつ能力とタスク t に必要な能力を比較し，ワーカ w が t に必要な能力をすべて有している場合に $ta(w)$ にタスク t を保存する関数
- *Worker_Available(w,t)*: ワーカ w のもつ能力とタスク t に必要な能力を比較し，ワーカ w が t に必要な能力をすべて有している場合に $wa(t)$ にワーカ w を保存する関数
- *Worker_Score(w,Formula)*: ワーカ w のスコアを現在の割り当て数から計算し，$score(w)$ に保存する関数
- *Score_Sort(score)*: $score$ を数値の高い順にソートし，上位 K 人のワーカを抽出し，$prior\_workers$ に保存する関数
- *Make_Graph(p)*: パス p における高優先度ワーカの最大割り当て数を算出するためのグラフを作成する関数（詳細は後述する）
- *Maximum_Flow(graph)*: $graph$ の最大流をフォード・ファルカーソン法により求める関数
- *Pworker_Allocation(graph)*: $Make\_Graph(p)$ で作成したグラフを利用し，パス p に対する高優先度ワーカの最大割当てを可能とする割当てを保存する
- *Generate_Candidate(allocation)*: $allocation$ の未割当てタスクに割当て可能なワーカの組を網羅的に求め，出力する関数
- *Limited_search(allocation_candidate)*: $allocation\_candidate$ を選択可能な割当てとして深さ制限探索を行い，スコアの合計値が最大となる割当てを求める関数
- *Worker_Count(w,allocation)*: $allocation$ における w の割り当て数を求める関数
- *Worker_Delete(w,wa,ta)*: wa と ta を参照し，ワーカ w の関係するハッシュを削除する関数

上述の関数について補足説明を加える．$Make\_Graph(p)$ において作成するグラフを $G = (V, E)$ とする．このとき，

$$V = prior\_workers \cup tasks_p \cup I \cup O$$

である．ここで，$tasks_p$ は p 内に含まれるタスクの集合，I は入力ノード，O は出力ノードの集合を表す．また，

$$E = I\_PW \cup PW\_T \cup T\_O$$

であり，

$$
\begin{aligned}
I\_PW &= I \times prior\_workers \\
PW\_T &= \bigcup_{w \in prior\_workers} (\{w\} \times (wa(w) \cap tasks_p)) \\
T\_O &= tasks_p \times O
\end{aligned}
$$

である．

ワーカの希望度を考慮した割当て手法のアルゴリズムを Algorithm 1 に示す．本アルゴリズムでは，7 行目では，ワークフロー内の全てのパスを P に保存，8 行目〜14 行目でワーカ w_i とタスク t_j に必要な能力を比較し w_i が実行可能なタスクのリスト $ta(w_i)$ と t_j を実行可能なワーカのリスト $wa(t_j)$ を作成している．15 行目〜47 行目ではループ処理を行っており，1 ループごとに P に属する 1 つのパスにワーカを割り当てている．ループ内では，17 行目〜19 行目でワーカ w_i の現割当てでのスコアを計算し $score(w_i)$ に保存，20 行目でそれをスコアの高い順にソートし上位 K 人を $prior\_workers$ に保存する．22 行目〜30 行目では P に属するパス p_i について，最大流問題を解くことで，p_i に $prior\_workers$ から割当てられるワーカの最大数を算出し，最も多くのワーカを $prior\_workers$ から割り当てることが可能な高優先度ワーカのみの割当て集合 $pw\_allocation$ を作成する．続いて，32 行目から 34 行目で $pw\_allocation$ に属する各割当てについて，まだタスクの割り当てられていないタスクに高優先度ワーカ以外のワーカを wa を参照し，網羅的な組み合わせを $allocation\_candidate$ に登録する．35 行目から 46 行目では，可能な割当てが 1 つもない場合にはループを脱するために end_frag を 1 にする．それ以外の場合には，深さ制限探索を行い最もスコアの合計が高い割り当て $allocation\_new$ を $Allocation$ に保存している．また同時に，$allocation\_new$ を割当てたことによりワーカの希望度を超える場合にはワーカを W から削除し，以降そのワーカにはタスクが割り当てられることがないように処理を行う．

以下，提案アルゴリズムの計算量について述べる．$Workflow = (V, E)$ に対して，$V = \{v_1, v_2, \ldots, v_n\}$ とし，ノード v の出次数を $deg(v)$ と表す．$Workflow$ のパス p_i における j 番目のタスクを t_{ij}，t_{ij} に割当て可能なワーカの集合を W_{ij} とする．以下の定理が成り立つ．

定理２：もし以下の 2 条件が成り立つならば，提案アルゴリズムは多項式時間で動作する．ここで，c は定数とする．

$$\prod_i deg(v_i) \in \mathcal{O}((|V| + |E|)^c) \tag{1}$$

$$\max_i(\prod_j |W_{ij}|) \in \mathcal{O}((|V| + |E| + |W|)^c) \tag{2}$$

証明の概略：アルゴリズムの計算量を支配するのは 22 行目の for 文および 33 行目の $Generate\_Candidate()$ であり，条件 (1) より $Workflow$ のサイズの多項式で抑えられる

- 22 行目の for 文の繰り返し回数は P に属するパスの数であり，条件 (1) より $Workflow$ のサイズの多項式で抑えられる
- 33 行目の $Generate\_Candidate()$ は，パスにおける未割当てタスクに対して割当て可能なワーカを網羅的に割り当て

Algorithm 1 Allocation Algorithm

Input: 能力の集合 $Abilities$, ワーカの集合 W, ワークフロー $Workflow$, ワーカの希望タスク数 $MT(w)$, スコアの計算式 $Formula$, 深さ制限探索の設定深度 $Depth$, 高優先度ワーカの数 K, 補正値 $Correction\_value$

Output: Allocation

Initialisation :

1: $Allocation = \emptyset$
2: $ta = \emptyset$
3: $wa = \emptyset$
4: $allocation\_num = \emptyset$
5: $score = \emptyset$
6: $end\_frag = 0$
7: $P = All\_Passes(Workflow)$
8: **for** $i = 1$ to $|W|$ **do**
9: $allocation\_num(w_i) = 0$
10: **for** $j = 1$ to $|T|$ **do**
11: $Worker\_Available(w_i, t_j)$
12: $Task\_Available(w_i, t_j)$
13: **end for**
14: **end for**

LOOP Process

15: **while** $end\_frag == 0$ **do**
16: $provisional\_allocation = \emptyset$
17: **for** $i = 1$ to $|W|$ **do**
18: $Worker\_Score(w_i, Formula)$
19: **end for**
20: $Score\_Sort(score)$
21: $maximum = 0$
22: **for** $i = 1$ to $|P|$ **do**
23: $graph = Make\_Graph(p_i)$
24: **if** $Maximum\_Flow(graph) > maximum$ **then**
25: $maximum = Maximum\_Flow(graph)$
26: $pw\_allocation = Pworker\_Allocation(graph)$
27: **else** $\{Maximum\_Flow(graph) == maximum\}$
28: $pw\_allocation.push(Pworker\_Allocation(graph))$
29: **end if**
30: **end for**
31: $allocation\_candidate = \emptyset$
32: **for** $i = 1$ to $|pw\_allocation|$ **do**
33: $allocation\_candidate.push(Generate\_Candidate(pw\_allocation_i))$
34: **end for**
35: **if** $|allocation\_candidate| == 0$ **then**
36: $end\_frag = 1$
37: **else**
38: $allocation\_new = Limited\_search(Depth, allocation\_candidate)$
39: $Allocation.push(allocation\_new)$
40: **for** $i = 0$ to $|W|$ **do**
41: **if** $Worker\_Count(w_i, allocation\_new) + allocation\_num(w_i) \geqq MT(w_i)$ **then**
42: $W.delete(w_i)$
43: $Worker\_Delete(w, wa, ta)$
44: **end if**
45: **end for**
46: **end if**
47: **end while**
48: **return** $Allocation$

ており，この数は条件 (2) より $Workflow$ のサイズとワーカ数の多項式で抑えられる

- $While$ 文の繰り返し回数の最大値は,「タスクが最小のパスのタスク数」で「全ワーカの希望タスク数の総計」を割った値である

以上より，条件 (1) および (2) が成り立つ場合，アルゴリズムは多項式時間で動作する．$(Q.E.D)$

なお，条件 (1) および (2) が成り立つ場合でも，定理 1 の NP 困難性は成り立つ．

通常のワークフローにおいて，条件 (1) が成立しないほどノードの出次数が多く複雑な構造をしたものは稀であると考えられる（例えば，次章の評価実験で用いる 2 つのワークフローはこの条件を満たす）．一方，条件 (2) については，タスクに割当て可能なワーカの数に依存しており，場合によってはアルゴリズムをより効率化することも必要と考えられる．この点については，さらなる検討が必要である．

5 評価実験

5.1 評価実験の概要

本節では，評価実験の内容について説明する．評価実験では，提案手法がワーカの希望度に沿った割当てを実現しているかどうかを評価する．また，プロジェクトへの参加を希望しているにも関わらず，タスクが割当てられないワーカが存在しないかどうかを同時に評価する．比較対象は，本研究と同様にワーカ側の事情に配慮した割当て手法であるインクルージョン性とワーカの能力に配慮した割当て手法 [1] を使用する．

評価実験には 2 種類のワークフローを使用する．これは関連研究 [1] のシミュレーションに用いられたワークフローと同一のものである．使用したワークフローは以下の通りである．

- ワークフロー A

関連研究 [1] で使用されている「説明用ワークフロー」と同一のものである．図 9 にこれを示す．図 9 中の $\{a_1, a_2, a_3, a_4\}$ は架空の能力であり，各タスクを表すノード内 {} が各タスクに必要な能力を表す．

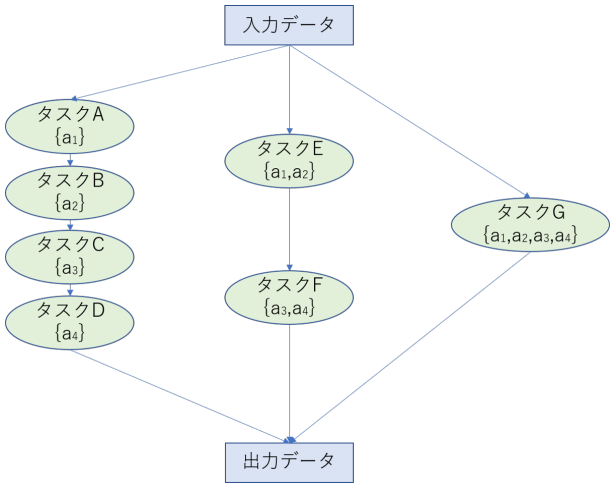

図 9　ワークフロー A

- ワークフロー B

関連研究 [1] で使用されている「手話交流ワークフロー」と同一のものである．図 10 にこれを示す．図中の能力について，

- JSL:日本手話を使用できる
- J:日本語を使用できる
- E:は英語を使用できる
- ASL:アメリカ手話を使用できる

である．また，このワークフローにおける入力データは「日本手話」，出力データは「アメリカ手話」であり,「日本のろう学生とアメリカのろう学生が手話での交流をする際に」[1] 使用されたものである．

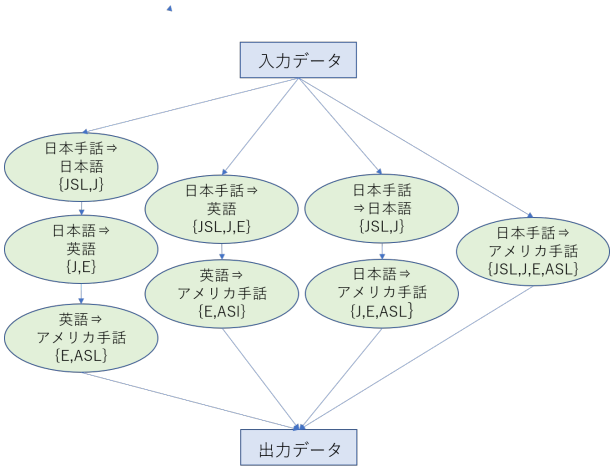

図 10　ワークフロー B

評価実験では分布を変化させた仮想のワーカ 100 人からなるワーカ群 W を生成し，提案手法と比較手法 [1] を適用する．本実験で変化させたのは，ワーカのもつ能力数の分布，ワーカの希望タスク数の分布の 2 点である．また，ワーカの能力数により希望タスク数に偏りがある場合を評価するために，ワーカの希望タスク数の分布はワーカのもつ能力数ごとに変化させた．この際，ワーカの希望タスク数 $n(n \in \mathbb{N})$ は $1 \leqq n \leqq 5$ の範囲内で確率的に与えられるように設定した．設定した確率の分布は $normal$，$small$，$large$ の 3 通りあり，内訳を表 1（単位は%）に示す．

表 1　タスク希望度の確率分布

分布名 $\backslash MT(w)$	1	2	3	4	5
normal	20	20	20	20	20
small	30	30	20	10	10
large	10	10	20	30	30

また，作成するワーカ群 W の能力数の分布を表 2（ワークフロー A）と表 3（ワークフロー B）に，ワーカの能力数ごとに設定した希望タスク数 $MT(w)$ の確率分布を表 4（ワークフロー A）と表 5（ワークフロー B）に示す．ただし，ワークフロー B において能力数 1 であるワーカや，能力数 2 であるが所持する能力が {JSL,ASL} であるようなワーカはタスク割当てを行うことができない．そのため，そのような能力を持つワーカは募集の段階で除外されていると想定し，ワーカ群 W はそ

れ以外の能力を該当する個数有している集合として作成する．

評価実験では，ワークフロー A に対して分布 (1)〜(4) と分布 (a)〜(d) を網羅的に組み合わせた 16 通り，ワークフロー B に対して分布 (5)〜(8) と分布 (e)〜(h) を網羅的に組み合わせた 16 通りのワーカ集合 W を作成する．(以降，作成したワーカ集合は 1_a のように表記する．) 作成した各ワーカ集合とワークフローの組を入力として提案手法を適用し，その結果を評価する．評価指標として使用するのは 2 章で述べた *inclusion* と *dissatisfaction_rate* である．また，本評価実験では $K = 25$, $Depth = 2$, $Correction\_value = 10$ とした．

表 2　ワーカの能力数分布（ワークフロー A）

分布名 \ 能力数	1	2	3	4
分布 (1)	25 人	25 人	25 人	25 人
分布 (2)	10 人	10 人	10 人	70 人
分布 (3)	70 人	10 人	10 人	10 人
分布 (4)	10 人	40 人	40 人	10 人

表 3　ワーカの能力数分布（ワークフロー B）

分布名 \ 能力数	1	2	3	4
分布 (5)	0 人	34 人	33 人	33 人
分布 (6)	0 人	10 人	10 人	80 人
分布 (7)	0 人	80 人	10 人	10 人
分布 (8)	0 人	10 人	80 人	10 人

表 4　タスク希望度の能力数別分布（ワークフロー A）

分布名 \ 能力数	1	2	3	4
分布 (a)	normal	normal	normal	normal
分布 (b)	large	large	small	small
分布 (c)	small	small	large	large
分布 (d)	small	large	large	small

表 5　タスク希望度の能力数別分布（ワークフロー B）

分布名 \ 能力数	1	2	3	4
分布 (e)	-	normal	normal	normal
分布 (f)	-	large	normal	small
分布 (g)	-	small	lnormal	large
分布 (h)	-	small	large	small

5.2　評価実験の結果

以下に評価実験の結果を示す．表 6 から表 13 は各ワーカ群とワークフローの組に対して，ワーカのインクルージョン性とリクエスタの事情双方に配慮した手法 [1]（以下の表においては，関連手法と記載する）と提案手法を適用した場合の *dissatisfaction_rate* を示したものである．また，本評価実験において *inclusion* はタスクの割り当てられていないワーカの有無を確認するために使用する指標であるが，関連研究の手法 [1] と提案手法を適用した結果，いずれの手法を適用した場合においても，全てのワーカ群とワークフローの組に対して $inclusion = 0$ であった．よって，本節で示す表と図からは省略している．図 11 と図 12 はワークフロー A とワークフロー B のそれぞれについて表に示した結果をまとめたものであり，横軸が関連手法，縦軸が提案手法の *dissatisfaction_rate* を表している．以下，前者を *dissatisfaction_rate_r*，後者を *dissatisfaction_rate_p* と呼称する．また，各点は 1 つのワーカ群とワークフローの組に対して両手法を適用した結果に対応する．図中の網掛け部は関連手法に比べ提案手法の *dissatisfaction_rate* が低い領域を示しており，$dissatisfaction\_rate\_r > dissatisfaction\_rate\_p$ であり，境界は $dissatisfaction\_rate\_r = dissatisfaction\_rate\_p$ である．

図 11 および図 12 から，全てのワーカ群とワークフローの組を入力とした場合において，$dissatisfaction\_rate\_p < dissatisfaction\_rate\_r$ であることがわかる．このことから，希望タスク数をベースにしたワーカの不満度は，関連手法を適用した場合と比較して，提案手法を適用した場合の方が低いことが示された．これは，手法の位置づけから明らかではあるが，提案手法がワーカの希望タスク数をワーカの希望と想定し入力データの一つとして使用しているのに対し，関連手法では希望タスク数は考慮の外にあり使用しないためであると考えられる．また前述の通り，ワーカ群とワークフローの組に提案手法を適用した全ての場合において，$inclusion = 0$ であった．これは *Correction_value* が有効に作用し，タスク未割当てのワーカのスコアは高く計算され，優先的にタスクを割り当てることが実現されたためであると考察される．以上から，提案手法は Inclusion を損なうことなく，Consideration を高めるという研究目的に対して有効な手法であると考えられる．

表 6　ワークフロー A, 分布 (a) での結果

手法 \ ワーカの分布	1_a	2_a	3_a	4_a
提案手法	0.092	0.088	0.100	0.092
関連手法	0.593	0.649	0.695	0.681

表 7　ワークフロー A, 分布 (b) での結果

手法 \ ワーカの分布	1_b	2_b	3_b	4_b
提案手法	0.093	0	0.115	0.091
関連手法	0.641	0.641	0.519	0.651

表 8　ワークフロー A, 分布 (c) での結果

手法 \ ワーカの分布	1_c	2_c	3_c	4_c
提案手法	0.083	0	0.081	0.093
関連手法	0.791	0.705	0.662	0.665

表 9　ワークフロー A, 分布 (d) での結果

手法 \ ワーカの分布	1_d	2_d	3_d	4_d
提案手法	0	0	0.107	0
関連手法	0.736	0.525	0.520	0.736

表 10　ワークフロー B, 分布 (e) での結果

手法 \ ワーカの分布	5_e	6_e	7_e	8_e
提案手法	0.131	0.126	0.124	0.114
関連手法	0.654	0.644	0.776	0.699

表 11　ワークフロー B, 分布 (f) での結果

手法 \ ワーカの分布	5_f	6_f	7_f	8_f
提案手法	0.127	0.114	0.143	0.128
関連手法	0.661	0.602	0.570	0.567

表 12　ワークフロー B, 分布 (g) での結果

手法 \ ワーカの分布	5_g	6_g	7_g	8_g
提案手法	0	0.143	0.115	0.114
関連手法	0.669	0.529	0.800	0.747

表 13　ワークフロー B, 分布 (h) での結果

手法 \ ワーカの分布	5_h	6_h	7_h	8_h
提案手法	0.104	0	0.098	0.142
関連手法	0.715	0.661	0.702	0.690

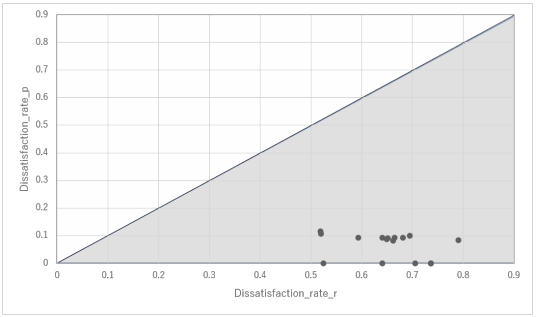

図 11　dissatisfaction_rate の比較（ワークフロー A）

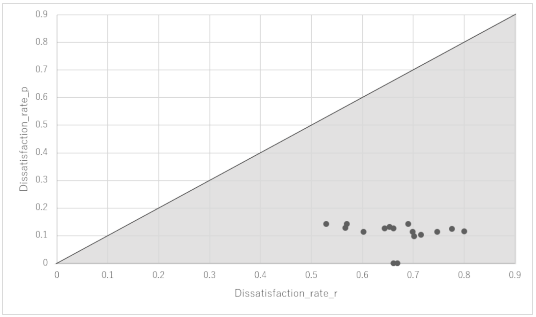

図 12　dissatisfaction_rate の比較（ワークフロー B）

6 むすび

本稿では，マイクロタスク型クラウドソーシングにおけるワーカの事情に配慮した新たな割り当て手法として，ワーカの希望タスク数を考慮したタスク割当て手法を提案し，シミュレーションを通して，その有用性の検証を行った．また本手法を提案するにあたり，ワーカの希望タスク数を考慮した割当てが NP 困難性をもつことも証明した．その結果，提案手法がワーカの希望タスク数に沿った形でタスクを割り当てることを実現していることを確認した．また，関連研究 [1] において必要性の指摘されていたインクルージョン性に則した観点から，全てのワーカに対して最低限何らかのタスクが割り当てられていることも確認した．

今後の課題としては，ワーカ側に配慮する事項を拡張することが第一に挙げられる．具体的にワーカの満足度をより高めるための事項として，ワーカが希望するタスクに優先的に割り当てる，逆にワーカがやりたくないと考えるタスクに対しては割当てを行わないといった事項を想定している．また，提案手法においては，関連研究 [1] とは異なり，生産性やスループット等のリクエスタ側の事情に関しては特別な配慮を行っていない．この点に関して，特にソーシャルインクルージョンの場面においてはリクエスタ側の事情は無視することのできない事項であり，改良する必要がある．

以上を優先的に実現するための今後の展望として，深さ制限探索をより有効に活用することが考えられる．本研究において深さ制限探索を導入した割当てを行ったが，本研究の範囲ではその利点が十分に生かされているとはいえない．スコアの算出により多くの要素を組み込む，アルゴリズムを簡略化し深さをより大きな値に設定可能にする等の改良を行うことで，より効果的な運用が可能になると考えている．また，深さ制限探索は設定深度が大きいほど処理にかかる時間が指数的に増加する特徴があり，評価実験中のワーカ群とワークフローの組においても処理時間が長くなる組が存在した．この点からもアルゴリズムを改良する余地がある．

7 参考文献

文　献

[1] A Task Assignment Method Considering Inclusiveness and Activity Degree, Hashimoto,H.; Matsubara,M.; Shiraishi,Y.; Wakatsuki,D.; Zhang,J.; Morishima,A. IEEE, Dec 2018, p. 3498-3503.

[2] Sihem Amer-Yahia, Senjuti Basu Roy. Toward Worker-Centric Crowdsourcing. Bulletin of the Technical Committee on Data Engineering. 2016, vol. 39, no. 4, p. 3-13.

[3] U. Pferschy and J. Schauer, The maximum flow problem with disjunctive constraints, Journal of Comb. Optim. (2013), 26:109-119

[4] 西 記代子. MLA 機関によるクラウドソーシングの活用について : 欧米の事例と日本における導入の可能性 (小特集 アーカイブズ・ノート). 国文学研究資料館紀要. アーカイブズ研究篇 = The bulletin of the National Institute of Japanese Literature. 人間文化研究機構国文学研究資料館 編. 2018, no. 14, p. 103-112.

[5] 井川 甲作, 比嘉 邦彦. 日本におけるマイクロタスク型 クラウドソーシング市場の現状調査 Asurvey study on the micro tasktype crowdsourcing market in Japan.

[6] 総務省. ”ICT による多様な人材の労働参加促進: クラウドソーシングの広がり”. 情報通信白書. https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h30/html/nd164420.html, (accessed 2020-7-15).