論　文

専門家チームの同時形成における制約充足問題の解法

山本　亮太[†]　　岡本　一志[†a)]

An Algorithm for Solving Constraint Satisfaction Problems in Concurrent Formation of Expert Teams

Ryota YAMAMOTO[†] and Kazushi OKAMOTO[†a)]

あらまし　複雑なタスクに対する専門家チームの形成には，スキルレベルや予算，チームサイズ要件を満たすことが望ましい．更に，チーム形成を要するタスクが複数存在する場合，専門家が複数のチームに割り当てられるべきではない．本研究では，複数のタスクに対してこれらの要件を全て満たすチーム（実行可能解）を同時に形成する手法を提案し，チームを一つずつ順に形成する場合と比較している．シミュレーション実験では，仮想的なチーム形成問題を複数生成し，制限時間を設けた上で各手法を実行し，実行可能解を得られた問題数を比較する．また，組織内でのチーム形成と，クラウドソーシングなどの組織外でのチーム形成を想定した問題のデータセットをそれぞれ生成している．実験の結果，組織内のチーム形成では制限時間が 1 秒程度，組織外のチーム形成では 200 秒以上のとき提案法が有効であることを確認している．制限時間を 2 時間としたとき，組織内のチーム形成では，解を出力した問題数が比較手法よりも約 20% 多く，組織外のチーム形成では約 10% 多くなっている．分析より，専門家集合のサイズや予算が小さいほど提案法の有効性が高くなることが示唆されている．

キーワード　専門家チーム形成，組織，クラウドソーシング，制約充足問題，メタヒューリスティクス

1. まえがき

特定のタスクに対して，スキルや経験をもった専門家の集合からタスクの成功に最適なチームメンバーを選択する問題を専門家チーム形成問題という [1]．この問題は，組織内のプロジェクト（タスク）の人員配置 [2]〜[5] だけでなく，クラウドソーシングなどでの外部の専門家の選出 [6]〜[11] にも適用できる．また，専門性を伴う複雑なタスクを遂行するクラウドソーシングでは，異なる専門性や能力をもった専門家によるチーム形成の必要性が指摘されている [12]．

本研究では，あるタスクに対して一つのチームを形成する問題を STF（Single Team Formation）問題とよぶ．STF 問題は，スキルや雇用コストなどに関する制約条件や目的関数をもつ最適化問題として解かれることが多い [1], [5]〜[8], [13]〜[21]．更に，STF 問題を適用するタスクが複数同時にあり，複数の STF 問題を同時に解く問題を MTF（Multiple Team Formation）問題とよぶ．MTF 問題では，1 人の専門家が複雑なタスクを複数受けもつことは現実的ではなく，形成された複数のチーム間で専門家が重複することは避ける必要がある．つまり，MTF 問題では，各タスクにおける STF 問題の制約条件に加えて，専門家が割り当てられるチームは最大で一つという重複制約を設けられる．

MTF 問題の解法では，タスクごとに順にチームを形成するアプローチ（OF：Ordered Formation）と複数のチームをまとめて同時に形成するアプローチ（CF：Concurrent Formation）の二つが考えられる．前者では，後に形成されるチームが比較的少数の専門家集合から抽出されることから，STF 問題の制約条件を全て満たすチーム（実行可能解）の発見が難しくなるのに対し，後者では，全てのタスクに対する実行可能解が得られる可能性が高くなることが予想される．

MTF 問題の中で，特に，スキル・雇用コスト・チームサイズの 3 要件をもつ STF 問題を同時に解くことを考える．このような MTF 問題では，重複制約を加えた計 4 種類の制約条件があり，実行可能解を求めるこ

[†] 電気通信大学大学院情報理工学研究科，調布市
Graduate School of Informatics and Engineering, The University of Electro-Communications, 1–5–1 Chofugaoka, Chofu-shi, 182–8585 Japan
a) E-mail: kaushi@uec.ac.jp
DOI:10.14923/transinfj.2023JDP7006

とが困難な場合がある．また，メタヒューリスティクスや分枝限定法などの最適化手法の中には事前に実行可能解を得ることが望ましいものがある [22]．そこで，本研究では，この MTF 問題の制約充足問題（MTF-4 問題）の解法として MTF-4 アルゴリズム，前述の 3 要件をもつ STF 問題の制約充足問題（STF-3 問題）の解法として STF-3 アルゴリズムを開発する．

開発する MTF-4 アルゴリズムは，CF アプローチに基づくものであり，まず，全てのタスクに対して STF-3 アルゴリズムを独立して適用する．このとき，複数のチームに割り当てられる専門家が生じるため，重複する専門家を取り除くように STF-3 アルゴリズムを繰り返し適用する．この MTF-4 アルゴリズムの開発にあたっては，タスクごとに順に STF-3 アルゴリズムを適用する OF アプローチとともに評価し，CF アプローチの妥当性を検証する．評価実験では，仮想的に生成した問題を用いるシミュレーション実験を行う．専門家集合のサイズやタスク数などの問題を構成する要素を問題生成パラメータとし，パラメータのグリッドサーチにより評価用問題を生成する．生成された問題に対して制限時間を課した上で各手法を適用し，実行可能解を出力できた問題数を比較する．

本論文の構成は以下のとおりである．**2.** では，専門家チームの形成問題の関連研究と本研究との相違点を述べる．**3.** では，MTF-4 問題を定式化し，本研究で開発する MTF-4 アルゴリズムの詳細を説明する．**4.** では，多次元ナップサック問題の解法を基礎とした STF-3 アルゴリズムを述べる．**5.** では，シミュレーション実験の環境とその実験結果を示す．

2. 関 連 研 究

2.1 専門家チームの形成問題

専門家チームの形成問題の適用場面は，大きく分けて，組織内でプロジェクトに参画する人員を選出する場合 [2]〜[5] と，クラウドソーシングなどの不特定多数の専門家の集合からチームメンバーを抽出する場合 [6]〜[11] がある．クラウドソーシングにおける専門家チーム形成では，仕事の規模や特性に応じて人材を雇用することが可能となっている [23]．本研究では，どちらの場面で適用してもロバストに実行可能解の有無を判定できるアルゴリズムを開発する．

専門家チームの形成問題の多くでは，ソーシャルネットワークに基づくチームメンバー間の協調性を目的関数として最適化問題を解いている [1], [13]〜[21], [24]．

表 1　先行研究における制約条件

制約条件	スキル	雇用コスト	チームサイズ	専門家の重複
Rangapuram+ [18]	○	○	○	×
Rahman+ [9]	○	○	△	×
Majumder+ [24]	○	×	×	○
Fitzpatrick+ [2]	○	×	×	○
Guterrez+ [3]	○	×	×	○
Yadav+ [10]	○	△	×	○
Yamamoto+ [11]	○	△	×	○

特に，Lappas らは，このような専門家チーム形成問題を初めて定式化し，スキル制約を，タスクの遂行に要する全スキルがチーム内のいずれかの専門家によって保有されていることとしている [1]．一方，Li らの研究では，必要なスキルについて，チーム内で保有している専門家の数に下限を設定している [13], [14]．より一般化されたスキル制約では，専門家が有するスキルをスキルレベルとして定量化し，チーム内の専門家のスキルレベルの総和に対しタスクの必須スキルレベルという下限が設けられている [15]〜[18]．

また，チーム内の雇用コストを考慮した研究もある [18]〜[20]．Rangapuram らはチーム内の雇用コストの総和が予算を下回る制約を設けているが [18]，Kargar らは雇用コストの総和を目的関数に加えている [19]．本研究では，タスクの予算はあらかじめ決まっていることを想定し，タスク完了までに要する専門家の雇用コストの総額が予算以下となる制約を設ける．

更に，形成するチームのサイズが大きすぎると，チーム内でコミュニケーションを取る相手が多くなりチームが分断される傾向が示唆されている [25]．Rangapuram らは，チームサイズが一定数以上大きくなるとチームのパフォーマンスが下がるとし，チームサイズに制限を課している [18]．

以上を踏まえ，本研究で扱う STF-3 問題はスキル要件と雇用コスト要件，チームサイズ要件を制約条件としてもつものとする．

2.2 既存研究における課題

表 1 に STF 問題や MTF 問題を扱った先行研究の一覧を示す．スキルや雇用コスト，チームサイズを制約条件とする STF 問題を扱った研究 [9], [18] があるが，これらは同時に複数のタスクに対してチームを形成することを想定しておらず，提案された解法を MTF-4 問題に適用することはできない．また，Rangapuram らの手法 [18] は，チーム形成の際の 3 要件に加えて，社会

的・地理的な距離に関する上限制約を設けており，協調性に関する目的関数を最適化するアルゴリズムのため，STF-3 問題に適していない．Rahman らの手法 [9] では，一つのタスクに対して複数のチームを形成するため，STF-3 問題で想定する出力は得られない．

スキル制約をもつ MTF 問題に取り組んだ研究 [2], [3], [24] もあるが，雇用コストやチームサイズ要件を考慮しておらず，これらの解法を MTF-4 問題に適用することは難しい．スキルや雇用コストを考慮し MTF 問題を解く研究 [10], [11] もあるが，これらの問題では，各チームにおける雇用コストの総和に関する合計値を目的関数や制約条件として設けているため，タスクごとに予算が設定されている MTF-4 問題に適用することは難しい．

3. MTF-4 問題の定義とその解法

本節では，MTF-4 問題を制約充足問題として数理的に定義し，その問題に対するヒューリスティックな解法である MTF-4 アルゴリズムについて説明する．

3.1 MTF-4 問題の定式化

n 人の専門家集合と，m 個のタスク集合を考える．専門家 i がタスク j に割り当てられているとき 1，そうでないとき 0 となる 2 値変数を $x_{ij} \in \{0,1\}$ とおく．専門家 i はスキルレベルベクトル $\boldsymbol{s}_i = [s_{i1}, s_{i2}, \ldots, s_{il}] \in \mathbb{N}_0^l$，タスク j は必須スキルレベルベクトル $\boldsymbol{t}_j = [t_{j1}, t_{j2}, \ldots, t_{jl}] \in \mathbb{N}_0^l$ をもつとすると，スキル制約は

$$\sum_{i=1}^{n} s_{ik} x_{ij} \geq t_{jk} \quad \forall k \in [1,l],\ \forall j \in [1,m] \tag{1}$$

で表現できる．専門家 i の雇用コストが $c_i \in \mathbb{N}$，タスク j の予算が $b_j \in \mathbb{N}$ のとき，予算制約は

$$\sum_{i=1}^{n} c_i x_{ij} \leq b_j \quad \forall j \in [1,m] \tag{2}$$

となる．なお，全ての専門家の雇用コストをまとめたベクトルを $\boldsymbol{c}$，全てのタスクの予算をまとめたベクトルを $\boldsymbol{b}$ とする．一方で，チームサイズの上限が K_j と決まっているとき，チームサイズに関する制約は

$$\sum_{i=1}^{n} x_{ij} \leq K_j \quad \forall j \in [1,m] \tag{3}$$

で定式化される．専門家の重複に関する制約は

Algorithm 1 multiple team formation

Input: $K, \boldsymbol{c}, \boldsymbol{b}, S, T$

1: **for all** $j \leftarrow [1,m]$ **do**
2: $\quad G_j \leftarrow STF(\emptyset, \boldsymbol{t}_j, b_j)$
3: $\quad$ **if** $G_j = \emptyset$ **then return** $\emptyset$
4: $U_{immut} \leftarrow \emptyset$
5: **while** $True$ **do**
6: $\quad \boldsymbol{b}' \leftarrow [-1, \ldots, -1]$
7: $\quad$ **for all** $j \leftarrow [1,m]$ **do**
8: $\quad\quad M_j \leftarrow G_j \cap \bigcup_{j' \neq j} G_{j'}$
9: $\quad\quad$ **if** $M_j \neq \emptyset$ **then**
10: $\quad\quad\quad b'_j \leftarrow (b_j - \sum_{i \in G_j} c_i)/|M_j|$
11: $\quad j' \leftarrow \mathrm{argmax}_{j \in [1,m]} b'_j$
12: $\quad$ **if** $M_{j'} = \emptyset$ **then break**
13: $\quad G_{j'} \leftarrow reform\,team(j')$
14: $\quad$ **if** $G_{j'} = \emptyset$ **then return** $\emptyset$
15: **return** $\{G_1, G_2, \ldots, G_m\}$

$$\sum_{j=1}^{m} x_{ij} \leq 1 \quad \forall i \in [1,n] \tag{4}$$

で表現される．本研究では，式 (1) ~ (4) を全て満たす変数を決定する制約充足問題を解く．

3.2 部分問題への分割

MTF-4 問題で式 (4) の制約をいったん無視すると，各タスクに対して式 (1) ~ (3) の制約条件を満たすチームの形成をそれぞれ独立に実行すれば十分である．一つのタスクに対して上記の 3 制約のもとでチームを形成する問題を STF-3 問題と呼ぶ．タスクごとに独立に STF-3 問題を解くと，$\sum_{j=1}^{m} x_{ij} > 1$ となるような，複数のチームに同時に割り当てられる専門家 i が生じることがある．このような専門家を重複ワーカと呼ぶ．MTF-4 アルゴリズムでは，重複ワーカ i の変数ベクトル $[x_{i1}, x_{i2}, \ldots, x_{im}]$ の一部を 0 に固定し $\sum_{j=1}^{m} x_{ij} = 1$ とした上で，再度タスクごとに STF-3 問題を解く．このような操作を式 (4) を満たすまで繰り返す．

3.3 MTF-4 アルゴリズム

3.3.1 メインアルゴリズム

本研究で提案するアルゴリズムの擬似コードを Algorithm. 1 に示す．入力の S, T はそれぞれ専門家とタスクのスキルレベルベクトルの集合である．このアルゴリズムでは STF-3 問題を解く関数 STF を用いているが，この処理の詳細は 4 節で説明する．

Algorithm. 1 では，まず，各タスクに独立して STF を実行する（1 ~ 4 行目）．STF でチーム形成が不可能の場合，主問題でも解が得られないため $\emptyset$ を返す．このように各タスクに独立してチーム形成した場合，重

Algorithm 2 reform team

Input: j

$U_{rm} \leftarrow \bigcup_{j'} G_{j'} \setminus G_j$
$u' \leftarrow -\infty, a' \leftarrow 2$
for all $u \leftarrow M_j$ **do**
  $a \leftarrow |\{\forall j' \in [1,m] \mid u \in G_{j'}\}|$
  $U_{rm} \leftarrow U_{rm} \cup \{u\}$
  $G \leftarrow STF(U_{rm}, \boldsymbol{t}_j, b_j)$
  if $G \neq \emptyset \wedge ((u' \notin U_{immut} \wedge u \in U_{immut}) \vee (a > a') \vee (a = a' \wedge \bigcup_{i \in G_j} c_i < \bigcup_{i \in G'_j} c_i))$ **then**
    $G' \leftarrow G,\ u' \leftarrow u,\ a' \leftarrow a$
  $U_{rm} \leftarrow U_{rm} \setminus \{u\}$
if $G' = \emptyset$ **then**
  if $G_j \cap U_{immut} = \emptyset$ **then**
    $U_{immut} \leftarrow U_{immut} \cup G_j$
    $G \leftarrow G_j$
  else return $\emptyset$
return G

複ワーカが発生する．以降の処理で，重複ワーカをもつチームに STF を繰り返し実行し，重複ワーカを取り除く．M_j はチーム j 内の重複ワーカ集合を示している．再形成するチームは，重複ワーカの数 $|M_j|$ に対する残り予算 b'_j が最も大きいチームとする．関数 $reform$ は，Algorithm. 2 に示され，$|M_j|$ が減少するようにチーム j に再形成を実行するアルゴリズムである．この処理を繰り返すことで，全てのチームの重複ワーカを取り除く．

3.3.2 チームの再形成

チームの再形成アルゴリズムの出力は，M_j 内のいずれかの重複ワーカが除去されたチームである．除去する重複ワーカを決定するために，重複ワーカのそれぞれが除外された場合の STF の結果を得る．

STF により得られた複数の解のうち，優先順位：1. 除外する重複ワーカが U_{immut} に含まれている，2. 除外する重複ワーカが割り当てられているチームの数がより大きい，3. 新たな解による雇用コストの総和がより小さい，を基準として出力するチームを選択する．U_{immut} とは，特定のチームに割り当てられることが確定しているワーカの集合である．

3.4 MTF-4 アルゴリズムの計算量と正当性

STF-3 アルゴリズムの最悪計算量を $O(2^n)$ とし，簡略化のため，$K = \max_{j=1,\cdots,m} K_j$ とする．このとき，MTF-4 アルゴリズムの最善計算量は，m 回独立して STF を適用し得られた初期のチーム間で専門家の重複がなかった場合で評価でき，$O(m \cdot 2^n)$ である．最悪計算量は，形成された初期の全チームで所属する全ての専門家が重複している場合で評価できる．これは，mK 人の専門家が重複していることに相当する．Algorithm. 1 の while ループの処理回数は専門家の重複数に依存し，このループ内で，重複ワーカについて一人ずつ割り当てられるチームを確定させていくため，関数 $reform\ team$ は $(m-1)K$ 回呼び出される．また，Algorithm. 2 より，$reform\ team$ 内で STF は最大 K 回実行される．したがって，MTF-4 アルゴリズムの最悪計算量は，$O((mK) \cdot (K \cdot 2^n)) = O(mK^2 \cdot 2^n)$ となる．MTF-4 問題に汎用ソルバーである SATisfiability problem Solver（SAT ソルバー）(注1) を適用する場合，最悪計算量は $O(2^{n \cdot m})$ であることから，m が大きくなるにつれ，最悪計算量の観点で，MTF-4 アルゴリズムは計算量を削減できるといえる．

MTF-4 アルゴリズムにおいて，必ずしも残り予算が大きいチームの方が再形成に成功しやすいとは限らなく，最終的に実行可能解を出力できるかどうかは再形成をするチームの順序にも依存する．また，代替候補となる専門家が重複ワーカのいない別のチームに存在する可能性があり，実行可能解を発見するために重複ワーカのいないチームの再形成を行う必要がある場合がある．よって，MTF-4 アルゴリズムでは，実行可能解が存在する問題に対して解なしと判定することがある．本研究では，このような出力を偽出力と呼ぶことにする．MTF-4 アルゴリズムは正当なアルゴリズムではないが，既存の正当なアルゴリズムと比較して計算時間が小さいため，制限時間がある場合は正しい解を出力する可能性が高くなることが期待される．

4. STF-3 アルゴリズム

本研究では，STF-3 問題を多次元ナップザック問題とみなし，Vasquez らの解法 [26], [27] を基本とする STF-3 アルゴリズムを提案する．Vasquez らの解法の中で，初期解の生成法と近傍探索のルールを変更する．

4.1 STF-3 問題と多次元ナップザック問題の類似性

チーム j の STF-3 問題では，最小化問題

$$min. \quad \sum_{i=1}^{n} c_i x_{ij} \tag{5}$$

$$s.t. \quad \sum_{i=1}^{n} s_{ik} x_{ij} \geq t_{jk} \quad \forall k \in [1,l] \tag{6}$$

$$\sum_{i=1}^{n} x_{ij} \leq K_j \tag{7}$$

(注1)：Google OR-tools, CP-SAT Solver など

Algorithm 3 single team formation

Input: $K, U_{rm}, S, \boldsymbol{t}, \boldsymbol{c}, b, z_{max}, z_{min}, z^*$
1: **if** $z_{max} = 0 \vee z_{min} > K$ **then return** $\emptyset$
2: **else** $z_{max} \leftarrow \min\{K, z_{max}\}$
3: $G \leftarrow \emptyset$, $q \leftarrow queue([z^*])$
4: **while** $|q| > 0$ **do**
5: $z \leftarrow q.pop()$
6: $\boldsymbol{x}^*_z \leftarrow relaxed\ solution\ at\ z$
7: **if** $f(\boldsymbol{x}^*_z) \leq b$ **then**
8: $G \leftarrow output\ of\ tabu\ search$
9: **if** $G = \emptyset$ **then**
10: **if** $z^* < z + 1 \wedge z + 1 \leq z_{max}$ **then**
11: $q.insert(z + 1)$
12: **else if** $z_{min} \leq z - 1 \wedge z - 1 < z^*$ **then**
13: $q.insert(z - 1)$
14: **else break**
15: **return** G

Algorithm 4 generate initial solution

Input: $U_{rm}, S, \boldsymbol{t}, \boldsymbol{c}, z$
1: $G' \leftarrow \emptyset$
2: **for all** $i \leftarrow [1, \ldots, z]$ **do**
3: $i' \leftarrow 0$, $e' \leftarrow 0$, $c' \leftarrow \infty$
4: **for all** $i \leftarrow [1, n]$ **do**
5: **if** $i \notin G' \wedge i \notin U_{rm}$ **then**
6: $e \leftarrow 0$
7: **for all** $k \leftarrow [1, l]$ **do**
8: $e \leftarrow e + \min\{s_{ik}, \max\{0, t_k - \sum_{i \in G'} s_{ik}\}\}$
9: **if** $e > e' \vee (e = e' \wedge c_i < c')$ **then**
10: $i' \leftarrow i$, $e' = e$, $c' = c_i$
11: **return** G'

の解が $\sum_{i=1}^{n} c_i x_{ij} \leq b_j$ を満たすことを確認することで実行可能解の有無を判定することができる．

Vasquez らは，多次元ナップザック問題を

$$max. \sum_{i=1}^{n} p_i x_i \quad s.t. \sum_{i=1}^{n} w_{ij} x_i \leq c_j \quad \forall j \in [1, l]$$

と定義している．p_i はアイテム i による利得，w_{ij} はアイテム i における資源 j の消費量，c_j は資源 j の総量を意味し，$p_i, w_{ij}, c_j \in \mathbb{N}$ である．Vasquez らの解法では，多次元ナップザック問題に等式制約 $\sum_{i=1}^{n} x_i = z$ を追加した問題の線形緩和解の周辺でタブーサーチを実行する．そして，あらゆる z について求めた解の中で最も目的関数値が高い解を最適解とする．

式 (6) を制約条件として式 (5) を最小化する問題は Vasquez らの多次元ナップザック問題に類似している．式 (6) に関して，両辺を正に保ったまま不等式の方向を逆転させることが困難なため，これらの問題の間で多項式時間還元を行うのは困難だが，近傍解を選ぶ基準を変更することで，それ以外は Vasquez らと同様のアルゴリズムで解けることが期待される．また，式 (7) の制約を満たすためには，Vasquez らの解法を $z \leq K_j$ の範囲で実行すればよいため，この解法を式 (5)〜式 (7) で示される最小化問題に適用することができる．

4.2 メインアルゴリズム

提案する STF-3 アルゴリズムの疑似コードを Algorithm. 3 に示す．z_{min} と z_{max} は，制約条件を式 (6) 及び $\sum_{i=1}^{n} c_i x_i \leq b$ とし，$\sum_{i=1}^{n} x_i$ を目的関数とする問題の線形緩和問題を解くことで決まる．線形緩和解を $\bar{\boldsymbol{x}}$ とすると，$z_{min} = \min(\lceil \sum_{i=1}^{n} \bar{x}_i \rceil), z_{max} = \max(\lfloor \sum_{i=1}^{n} \bar{x}_i \rfloor)$ となる．z^* は式 (6) を制約条件として式 (5) を最小化する問題の緩和解 $\boldsymbol{x}^*$ における $\sum_{i=1}^{n} x^*_i$ を四捨五入した値である．

1 行目においては，z_{max} が 0 若しくは z_{min} が K よりも大きい場合，実行不可能としている．3 行目以降は，z の値を変えつつ，緩和解を計算しタブーサーチを実行する処理を示している．なお，$f(\boldsymbol{x}) = \sum_{i=1}^{n} c_i x_i$ である．$z_{min} \leq z \leq min\{z_{max}, K\}$ の範囲において，$z^* \rightarrow z^* + 1 \rightarrow z^* - 1 \rightarrow z^* + 2 \rightarrow \cdots$ の順でタブーサーチを実行し，目的関数値が予算以下となるチーム G が見つかった時点で処理を終了する．

4.3 タブーサーチの初期解生成アルゴリズム

タブーサーチを実行する上で，初期解を設定する必要がある．Vasquez らは，緩和解 $[x_1, x_2, \ldots, x_n]$ において値が大きい順に選んだ z 個の変数を全て 1 にすることで初期解を生成している．また，Vasquez らの手法では，緩和解とのマンハッタン距離がしきい値以下となる解空間でタブーサーチを実行している．しかし，STF-3 問題において解空間を限定するとスキル制約を満たす解が見つかる保証がない．そこで本研究では，解空間を限定せずにスキル制約条件を全て満たしうる初期解を貪欲法で決定する．

初期解を生成する処理を Algorithm. 4 に示す．この処理はタスクに必要なスキルレベルを最も補完できる専門家から順に採用する方針でチーム形成をする．途中で必要なスキルをもつ専門家がいなくなるか，チームサイズが z となっても制約を満たせない場合は，その時点でのチームを返す．ここで，$is\_skill\_satisfied(G)$ は，G が式 (6) のスキル制約を満たすかを返す．

Algorithm. 4 でスキル制約を満たすチームを得られた場合，Algorithm. 5 を実行する．Algorithm. 5 は，Algorithm. 4 の出力に対して，可能な限り緩和解とのマンハッタン距離が小さくなるように，チーム内の専

Algorithm 5 approximate initial solution

Input: initial solution $G, S, \boldsymbol{t}, \boldsymbol{x}^*$
$G_r \leftarrow \{\forall i \in [1,n] \mid x_i^* > 0\}$
$G' \leftarrow G$
while True **do**
  $e' \leftarrow -\infty, G \leftarrow G'$
  for all $(u, u_r) \leftarrow G \times (G_r \setminus (G \cap G_r))$ **do**
    if $u \in G_r \vee u_r \in G$ **then continue**
    $G \leftarrow G \cup \{u_r\} \setminus \{u\}$
    if $is\_skill\_satisfied(G)$ **then**
      $e \leftarrow \sum_{k=1}^{l} \sqrt{\sum_{i \in G} s_{ik} - t_k}$
      if $e > e'$ **then**
        $e' \leftarrow e, G' \leftarrow G$
    $G \leftarrow G \cup \{u\} \setminus \{u_r\}$
  if $e' < 0$ **then break**
return G'

門家と，緩和解 $\boldsymbol{x}^*$ において $x_i^* > 0$ となる専門家の間で交換する操作を示している．ここでは，各スキルレベルの総和の必須スキルレベルに対する余りが最も大きくなる組み合わせに対して交換を実行する．この処理を交換後もスキル制約を満たす組み合わせがなくなるまで繰り返す．

4.4 タブーサーチにおける近傍探索

タブーサーチのアルゴリズムにおける基本的な操作は Vasques らの手法 [26] と同様である．しかし，STF-3 問題と多次元ナップサック問題の相違点や初期解生成アルゴリズムの変更により，近傍に遷移する際のルールが異なることになる．更に，探索の状況によってもそのルールは変化するため，後述の三つの探索の状況に対して近傍探索のルールを決めることとする．なお，$v(\boldsymbol{x}) = \sum_{k=1}^{l} \max\{0, t_{jk} - \sum_{i=1}^{n} s_{ik} x_i\}$ とする．

a) スキル制約を満たす解が見つかっていない場合

Algorithm. 4 での出力がスキル制約を満たしていない場合，制約を満たす解の探索を優先するため，緩和解との距離によって探索範囲を限定しない．近傍解のうち，優先順位：1.$v(\boldsymbol{x})$ がより小さい，2.$f(\boldsymbol{x})$ がより小さい，の基準に従って遷移する解を決定する．

b) 現在の解と緩和解との距離がしきい値を超える場合

a) には該当しないが，現在の解 $\boldsymbol{x}$ と緩和解 $\boldsymbol{x}^*$ とのマンハッタン距離 $\sum_{i=1}^{n} |x_i^* - x_i|$ がしきい値よりも大きい場合には，距離の小さい近傍解を優先する．このしきい値の計算方法は [26] と同様である．このとき，近傍解への遷移の際は優先順位：1.$v(\boldsymbol{x})$ がより小さい，2. 緩和解とのマンハッタン距離がより小さい，3.$f(\boldsymbol{x})$ がより小さい，の基準に従う．また，近傍解のうち，現在解よりも距離が大きくなる解は探索しない．

c) その他

a) にも b) にも該当しない場合，緩和解との距離が定義された値よりも小さい解の空間において優先順位：1.$v(\boldsymbol{x})$ がより小さい，2.$f(\boldsymbol{x})$ がより小さい，を基準として探索を実行する．

Vasquez らと同様に，解の遷移ごとにカウントされる $iter$ が，$v(\boldsymbol{x}) = 0$ となる解に至る前に上限 $iter_{max}$ を超えたときタブーサーチを終了する．また，$v(\boldsymbol{x}) = 0$ の解 $\boldsymbol{x}$ が見つかったとき $f(\boldsymbol{x}) \leq b$ であればその時点で STF-3 アルゴリズムを終了し，解を返す．

5. 評価実験

評価実験では，MTF-4 問題に対する OF アプローチと CF アプローチ，ベンチマークの汎用ソルバーによる性能を比較する．CF アプローチによる解法は，3 節で説明した MTF-4 アルゴリズムである．更に，OF アプローチと CF アプローチにおいて，STF-3 問題に汎用ソルバーを適用した際の性能も確認する．

5.1 実験方法

5.1.1 仮想的な MTF-4 問題の生成

本実験では，九つの問題生成パラメータにより仮想的な MTF-4 問題を生成する．各生成パラメータの候補値は表 2，表 3 のようにそれぞれ三つあり，各パラメータの値を定めることで一つの MTF-4 問題が生成される．実験では，全ての生成パラメータの組み合わせについて問題を生成し，3^9 個の問題を得る．そして，これらの問題に各手法を適用し，実行可能解を得られた問題数を比較する．

このシミュレーション実験では，組織内，組織外のチーム形成を想定した比較実験それぞれに候補値の設定を用意し，仮想問題を作成する．組織内と組織外のチーム形成問題では，問題生成のモデルには違いは無く，問題生成パラメータに違いがある．本実験では，開発した MTF-4・STF-3 アルゴリズムがどちらの場面に対しても有効性があることを検証する．組織内のチーム形成における実験の問題生成パラメータを表 2 に，組織外の場合の問題生成パラメータを表 3 に示す（詳細は 5.1.2 節で述べる）．

次に，問題生成パラメータにより MTF-4 問題を生成する方法を説明する．生成パラメータには，専門家数 n，タスク数 m に加えて，スキルの種類数 l も加える．複数チームを形成した際の専門家の重複度合いは l にも依存する．専門家 i のスキル k のレベル s_{ik} は，平均 μ，分散 σ^2 のベータ分布からサンプリングする

表 2　組織内チーム形成における問題生成パラメータ

パラメータ名	候補値
専門家数 n	50, 100, 200
タスク数 m	2, 5, 10
スキルの種類数 l	10, 20, 40
専門家のスキルレベル平均 μ	0.2, 0.5, 0.8
専門家のスキルレベル分散 σ^2	0.1, 0.25, 1
必須スキルレベルに関するパラメータ α	0.4, 0.5, 0.6
必須スキル数に関するパラメータ β	0.2, 0.4, 0.6
予算に関するパラメータ γ	0.6, 0.8, 1
チームサイズに関するパラメータ δ	0.6, 0.8, 1

表 3　組織外チーム形成における問題生成パラメータ

パラメータ名	候補値
専門家数 n	500, 1000, 2000
タスク数 m	5, 10, 20
スキルの種類数 l	10, 20, 40
専門家のスキルレベル平均 μ	0.2, 0.5, 0.8
専門家のスキルレベル分散 σ^2	0.1, 0.25, 1
必須スキルレベルに関するパラメータ α	0.1, 0.2, 0.3
必須スキル数に関するパラメータ β	0.2, 0.4, 0.6
予算に関するパラメータ γ	0.6, 0.8, 1
チームサイズに関するパラメータ δ	0.6, 0.8, 1

ものとし

$$s_{ik} \fallingdotseq 100r, \quad r \sim \mathrm{Beta}(a,b)$$
$$a = \frac{\mu(\sigma^2-\mu^2+\mu)}{\sigma^2}, \quad b = \frac{(1-\mu)(\sigma^2-\mu^2+\mu)}{\sigma^2}$$

のように算出する．本研究では，現実のクラウドソーシングサービスのデータを用いることができなかったため，実際のスキルレベルの分布を仮定することは難しく，多様な分布を表現可能なベータ分布を採用している．雇用コスト c_i は，専門家 i がもつスキルレベルの平均値をパラメータとするポアソン分布からサンプリングする．すなわち，

$$c_i \sim \mathrm{Poisson}\left(\sum_{k=1}^{l} s_{ik} \,/\, l\right).$$

一般に，労働市場での給与・報酬（インセンティブ）は，求める人材の能力の高さに応じて決まることが多く，本研究でもこの仮定をおくこととする．そのため，雇用コストをスキルレベルの大きさに相関するモデルとして，ポアソン分布を採用している．各タスクの必須スキルレベルは

$$t_{jk} = \left\lceil \alpha_j y_{jk} \sum_{i=1}^{n} s_{ik}/m \right\rceil, \ \alpha_j \sim N(\alpha, 0.1)$$
$$\boldsymbol{y}_j \in \{0,1\}^l, \ ||\boldsymbol{y}_j|| \fallingdotseq \max(\beta l, 1)$$

とし，各スキルは，各専門家がもつスキルレベルの合計値に相関させ，パラメータ α で大小を調整する．全てのタスクに関して α の値を固定すると，同一のスキルに関する必須スキルレベルが一意に定まってしまうため，タスクごとに α_j を決定する．α_j は平均を α，標準偏差を 0.1 とし，0 より大きい値を取る切断正規分布からサンプリングする．本実験では，各タスクにおいて必要となるスキルは一部であるとする．ベクトル $\boldsymbol{y}_j$ は，タスク j において各スキルを必要とするか否かを 2 値変数で示している．ノルム $||\boldsymbol{y}_j||$，すなわち，タスク j に要するスキル数はパラメータ β を用いて βl で表す．このスキル数は 1 以上である必要がある．また，必須スキルレベルの計算では，タスク数の増加による全体として必要となる専門家数の増大を防ぐために m で商をとる．予算 b_j と最大のチームサイズ K_j は

$$b_j \fallingdotseq \gamma \cdot \sum_{k=1}^{l} t_{jk} \,/\, \beta l, \ K_j \fallingdotseq \delta \cdot \frac{\sum_{k=1}^{l} t_{jk}/\beta l}{\mu}$$

とし，タスクの必須スキルレベルの平均値に相関させる．K_j の計算では，チームに必要な専門家数の期待値を求めるために，タスクの必須スキルレベルの平均値に対して専門家のスキルレベル平均 μ で商をとる．

式 (1) ~ (4) 内の各定数の値は全て整数値をとるため，サンプリングした結果に対してそれぞれ四捨五入する．なお，t_{jk} に関しては天井関数をとる．

5.1.2　問題生成パラメータの設定

表 2 と表 3 に記載の候補値の設定理由について述べる．まず，専門家数 n，タスク数 m，スキルの種類数 l は大きくしすぎると計算負荷が大きくなり実験が困難となる．クラウドソーシングを対象とした Liu ら [7] や Rahman らの研究 [9] では専門家数 n の上限を 3000〜5000 に設定しており，先行研究 [6]〜[10] ではスキルの種類数 l を 3，19，40，100 に設定している．本実験では，n と l の上限を先行研究にできるだけ近い値（$n = 2000$，$l = 40$）を採用することとし，また，タスク数 m は計算環境の都合上，20 を上限としている．なお，クラウドソーシングなどでの組織外のチーム形成に比べ，組織内のチーム形成では，専門家集合や同時に生じるタスク数が小さくなると想定されることから，相対的に n や m の上限を小さくしている．

パラメータ α は，専門家集合に対して，チーム形成を通じていずれかのタスクに割当られる専門家の割合を意味している．本実験で扱うチーム形成問題は，

(1) 組織内のチーム形成問題：組織内でプロジェクトに参画する人員を選出する場合で，比較的多くの専門家がいずれかのプロジェクトに参画するシナリオ，(2) 組織外のチーム形成問題：組織内の場合とは異なり，専門家集合の一部の専門家がタスクに割当られるシナリオ，をそれぞれ想定したものである．そのため，組織内の問題では $\alpha = 0.5$ 程度，組織外の問題では $\alpha = 0.1 \sim 0.3$ を採用している．パラメータ γ，δ は形成されるチームの総雇用コスト及びチームサイズの期待値を 1 としたときの割合をそれぞれ意味している．これは，γ で予算が十分か，δ で許容されるチームサイズの値が十分かといった要素を表現しており，γ，δ を小さすぎたり大きすぎたりする値に設定してしまうと簡単に解けるような問題が生成されることになる．実行可能解の有無が即座に判別でき，どのような解法でも計算時間や偽出力の可能性で差が出ないことになるため，本実験では，0.6 未満と 1 より大きな値は採用しないこととしている．また，パラメータ μ，β，σ^2 は，網羅性を高めるため，大中小の 3 種類の値を設定している．

5.1.3 実験設定

OF アプローチでは，予想されるチームサイズ $\frac{\sum_{k=1}^{l} t_{jk}/\beta l}{\mu_s}$ が大きいタスクから順に STF-3 問題を解く．実験の際，OF，CF アプローチにおける STF-3 問題で，**4.** で述べた STF-3 アルゴリズム tabu の他に，SAT ソルバーを適用する解法 solver を適用した結果も示す．OF アプローチまたは CF アプローチ（Algorithm 1, 2），tabu（Algorithm 3, 4, 5）または solver の組み合わせにより，4 通りの結果（OF+tabu, OF+solver, CF+tabu, CF+solver）が得られる．更に，MTF-4 問題に対して SAT ソルバーを適用する手法 benchmark の結果も同様に示す．solver や benchmark で用いる SAT ソルバーは，汎用ソルバーである Google OR-Tools(注2) の CP-SAT Solver とする．

MTF-4 問題に各手法を適用した際の出力は，実行可能解を発見できたときの「解あり」，実行可能解が存在しないと判定したときの「解なし」の 2 種類ある．問題によっては，それらの出力に多大な時間がかかる場合が想定されるため，本実験では，手法の実行の際に制限時間を設ける．$3^9 = 19683$ の問題に 5 種類の手法を適用する必要があり，利用可能な計算環境の都合上，本実験では 1 問題あたりの制限時間を 2 時間とし，制限時間以内に実行可能解の有無を判定できなかった場合は「タイムアウト」と出力する．また，タブーサーチにおいて $iter_{max} = 2000, iter_{min} = 500$ と設定する．**4.3** の線形計画緩和問題を解く上で，Google OR-Tools のソルバー GLOP を利用している．各手法は C++ で実装し，AMD EPYC 7713P 2.0 GHz の CPU，512 GB RAM の計算機上で実行している．

(注2)：https://developers.google.com/optimization

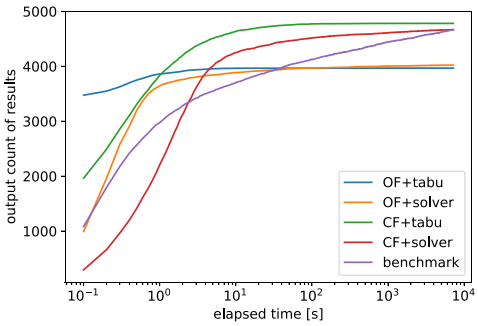

図 1　実行可能解の出力数の比較（組織内）

表 4　制限時間が 2 時間のときの各手法の出力数（組織内）

手法	解あり	解なし	タイムアウト	最小偽出力	最大偽出力
OF+tabu	3967	15716	0	998	1224
OF+solver	4021	15586	76	927	1170
CF+tabu	4781	14896	6	184	410
CF+solver	4666	14841	176	232	525
benchmark	4668	14492	523	-	-

5.2 実験結果

5.2.1 組織内の専門家チーム形成

まず，組織内の専門家チーム形成における実験の結果を示す．横軸を制限時間，縦軸を解ありの出力数とした片対数グラフを図 1 に示す．実験の際，1 問題あたりの制限時間を 2 時間としているが，2 時間以内に解けた問題の実行時間を記録することで制限時間を変化させたときの出力数を求めることができる．

表 4 は，制限時間を 2 時間としたときの出力数を種類別に示している．汎用ソルバーの benchmark は偽出力をしない正当なアルゴリズムであり，各手法の偽出力は benchmark が解ありと出力した問題に解なしと出力した問題数を数えることで求めることができる．しかし，benchmark がタイムアウトした問題では実行可能解の存在を断定できないため，断定できる偽出力の数を最小偽出力数，起こりうる最大の偽出力の数を最大偽出力数としてそれぞれ求める．最小偽出力数は表 4 の benchmark での解ありと出力した問題の集合と

各手法における解なしと出力した問題の集合の共通部分のサイズである．最大偽出力数は benchmark でタイムアウトを出力した問題が全て解あり，各手法でタイムアウトを出力した問題が全て解なしであると仮定したときの偽出力数である．最小偽出力数や最大偽出力数が小さいほど，制限時間を 2 時間よりも長くしたときの解ありの出力が多くなると予想できる．

図 1 と表 4 より，OF+tabu と CF+tabu を比較したとき，制限時間が 1 秒程度以下で OF+tabu が解ありの出力数において上回っており，タイムアウトの出力数は OF+tabu が小さくなっている．問題の解の出力にかかる実行時間は OF+tabu が小さい傾向がある．しかし，OF+tabu は偽出力数が大きく，制限時間が 1 秒程度以上で解ありの出力数が逆転している．また，制限時間が 2 時間のとき，CF+tabu は OF+tabu のよりも約 20% 多く解を出力している．OF+solver と CF+solver の比較でも同様の傾向が読み取れ，組織内のチーム形成においては CF アプローチが有効といえる．**3.4** で示したとおり，OF アプローチでは最悪計算量が $O(m \cdot 2^n)$ となり，CF アプローチの $O(mK^2 \cdot 2^n)$ よりも小さくなる．そのため，OF アプローチでは短い時間で解なしを出力するが，チームを再形成しないことから探索する解空間が小さくなってしまい，得られた出力が偽出力となる傾向が高いためと考えられる．

CF+tabu と CF+solver を比較すると，CF+tabu が制限時間によらず解ありの出力数で上回っており，偽出力数もより小さい傾向にある．tabu では，予算要件をいったん無視して解を得ており，その計算は少なくとも solver が解を出力するよりは早く求まる．そして，雇用コストの総和が予算を下回る解を探索するのに時間がかからない問題が多いためと考えられる．CF アプローチを適用する場合，STF-3 問題の解法として tabu が最も有効性が高いといえる．

CF+tabu と benchmark を比較すると，制限時間が大きいときは解ありの出力数の差が小さくなっている．benchmark は偽出力をしないことから，制限時間を大きくするといずれは CF アプローチよりも出力数を上回る．**3.4** で求めた最悪計算量では，CF+tabu が benchmark よりも小さくなっている．その一方で，本実験の組織内のチーム形成問題では，計算量が依存しているパラメータ n と m の値が小さく，計算時間の差が小さくなることが要因として考えられる．MTF-4 問題に対する結果を数時間以内に得る必要がある場合のみ CF+tabu を採用し，それよりも多大な時間を取れる場合は benchmark の採用が望ましい．

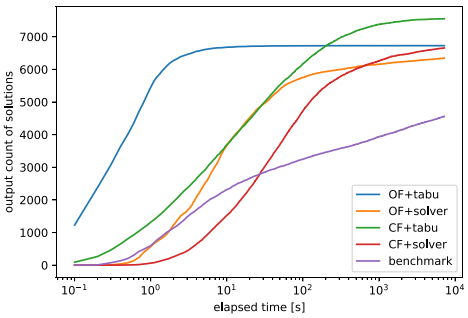

図 2　実行可能解の出力数の比較（組織外）

表 5　制限時間が 2 時間のときの各手法の出力数（組織外）

手法	解あり	解なし	タイムアウト	最小偽出力	最大偽出力
OF+tabu	6726	12955	2	846	1430
OF+solver	6345	11078	2260	444	1811
CF+tabu	7551	11993	139	13	605
CF+solver	6655	10998	2030	20	1501
benchmark	4559	11527	3597	-	-

5.2.2　組織外の専門家チーム形成

組織内のチーム形成と同様に，組織外チーム形成における実験の結果を図 2，表 5 に示す．図 2 及び表 5 のタイムアウトの出力数より，組織内のチーム形成と同様に，OF+tabu は CF+tabu よりも出力までの実行時間が短く，偽出力数が大きい傾向にあることがわかる．また，図 2 から，CF+tabu の出力数は制限時間が 200 秒程度で OF+tabu を上回り，制限時間が 2 時間のとき約 12% 多くの解ありを出力している．経過時間が大きくなるにつれ CF+tabu の出力数が OF+tabu を上回る理由は，組織内の問題と同様で，探索する解空間が小さくなるためと考えられる．CF+tabu の出力数が OF+tabu を上回る経過時間が組織内での結果と比較して大きくなっているのは，両者の計算量の差がパラメータ n と m の値の増加に伴い広がっているためと考えられる．OF+tabu のタイムアウトの出力数は 2 であるため，制限時間を 2 時間より大きくしたとき，OF+tabu の解ありの出力数は 6728 となる．つまり，制限時間が 2 時間より大きくしても，解ありの出力数において OF+tabu が CF+tabu を上回ることはない．OF+solver と CF+solver は CF+tabu と比較して，どちらもタイムアウトの出力数と偽出力数が大きく，制限時間によらず解ありの出力数が小さいことがわかる．benchmark もタイムアウトの出力数が大きく，制限時間が 2 時間のときの解ありの出力数が CF+tabu の約 60% となっ

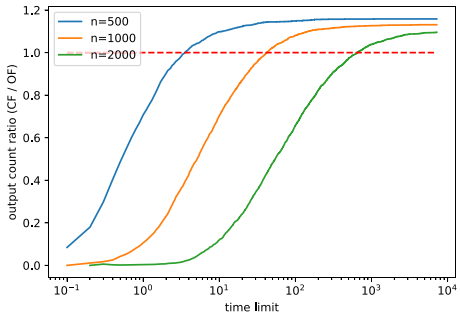

図 3　CF, OF アプローチにおける解の出力数の比較 - n

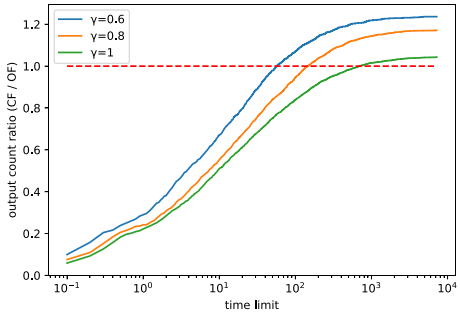

図 4　CF, OF アプローチにおける解の出力数の比較 - γ

ている．以上より，出力までの実行時間が数分程度かかることが許容されている状況では，CF+tabu が最も有効である．

図 3，図 4 は，専門家集合のサイズ，予算制約に関係する問題生成パラメータ n, γ について，候補値ごとに CF+tabu の OF+tabu に対する解ありの出力数の比率を示している．どちらも，CF+tabu の解ありの出力数はある時点を境に OF+tabu の出力数を上回っている．n が小さくなるほど，出力数が逆転する時点が小さくなり，$n = 2000$ のとき 500 秒程度で 1 を上回るが，$n = 500$ のときは 3 秒程度で上回る．これは，STF-3 問題を解く際に探索空間が小さくなり，実行時間が小さくなるためであると考えられる．また，γ が小さくなるほど最終的な CF+tabu と OF+tabu の出力数の比率が大きくなっている．これは，γ が小さくなるにつれ実行可能領域が小さくなり，OF+tabu の解が得られる可能性が低くなることが要因だと考えられる．タスクのスキルレベルに対して予算が少ないときほど，OF+tabu よりも CF+tabu を採用するべきといえる．

6. むすび

本研究では，複雑なタスクが同時に複数存在する場合の専門家チームの形成問題を MTF-4 問題として定式化している．MTF-4 問題に対する解法は，専門家の重複を考慮しつつ，タスクごとに順にチームを形成する OF アプローチと，全てのチームを同時に形成する CF アプローチに大別される．評価実験では，問題生成パラメータにより仮想的に生成される MTF-4 問題について，OF アプローチ，CF アプローチによる解法を制限時間を設けた上で適用し，性能を比較している．

組織内，組織外のチーム形成を想定した実験より，OF アプローチは CF アプローチよりも実行可能解の有無を判定するまでの計算時間は短いが，本来解が存在する問題に解無しと判定してしまう場合が多く，制限時間が大きい場合は CF アプローチが実行可能解を出力する可能性が高くなることを確認している．組織内でのチーム形成では 1 秒程度，組織外では 200 秒程度以上の実行時間を許容できる場合，CF アプローチが有効であることがわかった．更に，専門家集合のサイズや予算の値が小さいほど，CF アプローチの有効性が高くなると考えられる．また，STF-3 問題に対する解法において本研究で提案した tabu は汎用ソルバーよりも優れており，制限時間が 2 時間という設定において，どちらも本研究で提案した CF+tabu の組み合わせによる手法が最も多く実行可能解を出力する可能性が高いことがわかった．

今後の発展として，タスクの開始，終了を考慮した時系列的な分析が挙げられる．時系列要素を考慮すると，システムに追加されるワーカが増えるなどしてワーカプールの状態がチーム形成をするタイミングによって変わる可能性がある．このような状況で，一定期間に発注されたタスクをまとめてチーム形成するバッチ処理を行う際の CF アプローチと，タスクが受注されるごとにチームを形成する OF アプローチでどのような差が現れるのかを分析する．

謝辞　本研究は JSPS 科研費 JP21H03553, JP23H01635 の助成を受けたものです．

文　献

[1] T. Lappas, K. Liu, and E. Terzi, “Finding a team of experts in social networks,” Proc. 15th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, pp.467–476, 2009.

[2] E.L. Fitzpatrick and R.G. Askin, “Forming effective worker teams with multi-functional skill requirements,” Computers and Industrial Engineering, vol.48, no.3, pp.593–608, 2005.

[3] J.H. Gutiérrez, C.A. Astudillo, P.B. Pérez, D.M. Melià, and A.C. Véjar, “The multiple team formation problem using sociometry,” Computers and Operations Research, vol.75, pp.150–162, 2016.

[4] J. Zhang, P.S. Yu, and Y. Lv, “Enterprise employee training via project team formation,” Proc. 10th ACM Int. Conf. Web Search and Data Mining, pp.3–12, 2017.
[5] Y. Kou, D. Shen, Q. Snell, D. Li, T. Nie, G. Yu, and S. Ma, “Efficient team formation in social networks based on constrained pattern graph,” Proc. 2020 IEEE 36th Int. Conf. Data Engineering, pp.889–900, 2020.
[6] Y. Sun, W. Tan, and Q. Zhang, “An efficient algorithm for crowdsourcing workflow tasks to social networks,” Proc. 2016 IEEE 20th Int. Conf. Computer Supported Cooperative Work in Design, pp.532–538, 2016.
[7] Q. Liu, T. Luo, R. Tang, and S. Bressan, “An efficient and truthful pricing mechanism for team formation in crowdsourcing markets,” Proc. 2015 IEEE Int. Conf. Communications, pp.567–572, 2015.
[8] W. Wang, Z. He, P. Shi, W. Wu, Y. Jiang, B. An, Z. Hao, and B. Chen, “Strategic social team crowdsourcing: forming a team of truthful workers for crowdsourcing in social networks,” IEEE Trans. Mobile Computing, vol.18, no.6, pp.1419–1432, 2019.
[9] H. Rahman, S.B. Roy, S. Thirumuruganathan, S.A. Yahia, and G. Das, “Task assignment optimization in collaborative crowdsourcing,” Proc. 2015 IEEE Int. Conf. Data Mining, pp.949–954, 2015.
[10] A. Yadav, A.S. Sairam, and A. Kumar, “Concurrent team formation for multiple tasks in crowdsourcing platform,” Proc. 2017 IEEE Global Communications Conf., pp.1–7, 2017.
[11] R. Yamamoto and K. Okamoto, “Worker organization system for collaborative crowdsourcing,” Int. Conf. Production Research, 0093, 2021.
[12] A. Kittur, J.V. Nickerson, M. Bernstein, E. Gerber, A. Shaw, J. Zimmerman, M. Lease, and J. Horton, “The future of crowd work,” Proc. 2013 Conf. Computer Supported Cooperative Work, pp.1301–1318, 2013.
[13] C.T. Li and M.K. Shan, “Team formation for generalized tasks in expertise social networks,” Proc. 2010 IEEE 2nd Int. Conf. Social Computing, pp.9–16, 2010.
[14] C.T. Li, M.K. Shan, and S.D. Lin, “On team formation with expertise query in collaborative social networks,” Knowledge and Information Systems, vol.42, pp.441–463, 2015.
[15] C. Dorn and S. Dustdar “Composing near-optimal expert teams: A trade-off between skills and connectivity,” Proc. OTM Confederated Int. Conf. On the Move to Meaningful Internet Systems, pp.472–489, 2010.
[16] F. Farhadi, M. Sorkhi, S. Hashemi, and A. Hamzeh, “An effective expert team formation in social networks based on skill grading,” Proc. 2011 IEEE 11th Int. Conf. Data Mining Workshops, pp.366–372, 2011.
[17] K. Selvarajah, P.M. Zedeh, Z. Kobti, Y. Palanichamy, and M. Kargar, “A unified framework for effective team formation in social networks,” Expert Systems with Applications, vol.177, 114886, 2021.
[18] S.S. Rangapuram, T. Bühler, and M. Hein, “Towards realistic team formation in social networks based on densest subgraphs,” Proc. Int. Conf. World Wide Web, pp.1077–1087, 2015.
[19] M. Kargar, A. An, and M. Zihayat, “Efficient bi-objective team formation in social networks,” Proc. 2012th European Conf. Machine Learning and Knowledge Discovery in Databases, pp.483–498, 2012.
[20] M. Kargar, M. Zihayat, and A. An, “Finding affordable and collaborative teams from a network of experts,” Proc. 2013 SIAM Int. Conf. Data Mining, pp.587–595, 2013.
[21] L. Chen, Y. Ye, A. Zheng, F. Xie, Z. Zheng, and M.R. Lyu, “Incorporating geographical location for team formation in social coding sites,” World Wide Web, vol.23, pp.153–174, 2020.
[22] 久保幹雄，J.P. Pedroso, メタヒューリスティクスの数理，共立出版，東京，2009.
[23] 鹿島久嗣，小山　聡，馬場雪乃，ヒューマンコンピュテーションとクラウドソーシング，講談社，東京，2016.
[24] A. Majumder, S. Datta, and K.V.M. Naidu, “Capacitated team formation problem on social networks,” Proc. 18th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, pp.1005–1013, 2012.
[25] R. Kenna and B. Berche, “Managing research qualit: Critical mass and optimal academic research group size,” Management Mathematics, vol.23, no.3, pp.195–207, 2012.
[26] M. Vasquez and J.K. Hao, “A hybrid approach for the 0-1 multidimensional knapsack problem,” Proc. 17th Int. Joint Conf. Artificial Intelligence, pp.328–333, 2001.
[27] M. Vasquez and Y. Vimont, “Improved results on the 0-1 multidimensional knapsack problem,” European J. Operational Research, vol.165, no.1, pp.70–81, 2005.

（2023 年 2 月 11 日受付，9 月 18 日再受付，
12 月 1 日早期公開）

山本　亮太

2021 電気通信大学情報理工学域卒．2023 同大学大学院情報理工学研究科情報学専攻了．同年，Sansan 株式会社入社．現在に至る．修士（工学）．

岡本　一志　（正員）

2006 高知工科大学工学部情報システム工学科卒．2008 同大学院工学研究科基盤工学専攻修士課程了．2011 東京工業大学大学院総合理工学研究科知能システム科学専攻博士後期課程了．同年，千葉大学アカデミック・リンク・センター特任助教．2015 電気通信大学大学院情報理工学研究科総合情報学専攻助教．2020 同大学院情報理工学研究科情報学専攻准教授．現在に至る．博士（工学）．データサイエンスの研究に従事．