クラウドソーシングによるHuman-powered Joinの効率化

三津石智巳† 森嶋 厚行††,††† 品川 徳秀†† 青木 秀人††††

† 筑波大学大学院 図書館情報メディア研究科 〒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/知的コミュニティ基盤研究センター 〒305–8550 茨城県つくば市春日 1–2
††† JST さきがけ
†††† 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
E-mail: †tomomi@slis.tsukuba.ac.jp ††{mori,siena}@slis.tsukuba.ac.jp ††††hideto.aoki@klis.tsukuba.ac.jp

あらまし 群衆の知や力の利用により問題解決を行うクラウドソーシングシステム（CS システム）が数多く登場している．CS システムには，選択演算や結合演算など DB 操作に対応する処理がしばしば含まれており，これまで，このような演算の処理の効率化についていくつかの手法が提案されてきた．しかし，これらの手法では，必ずしも現実には成立しない次のような仮定が置かれてきた．（仮定 A）人手の作業は群衆の誰でも同様に行うことができる，（仮定 B）CS システムのプログラマ自身が演算対象に関する固有の知識を持つ．本稿では，これらの仮定が成立しない場合の Human-powered Join の処理を効率化するための処理効率化手法を提案する．

キーワード クラウドソーシング，結合演算

1. はじめに

計算機ネットワーク技術の発達に伴い，計算機だけではなく人も情報資源や処理装置とみなし，問題解決のためのシステムに含めることが可能になった．その結果，計算機には困難なデータ収集や処理を部分的に人手の作業によって行うシステムが数多く出現している．例えば，画像のタグを人手で付与する ESP Game [1] や，OCR で処理できない画像から人手でテキスト抽出を行う reCAPTCHA [2] といったシステムがある．これらのシステムは，クラウドソーシングシステム（以下，**CS システム**）と呼ばれている [3]．また，CS システムは，アプリケーションごとに個別のシステムとして開発されるだけでなく，CS システムに必要な機能を提供するクラウドソーシングプラットフォーム（以下，**CS プラットフォーム**）と呼ばれるシステム上に構築されることもある．例えば，Amazon Mechanical Turk（以下，**MTurk**）は，少額の謝金が支払われる作業を群衆に割り当てるための市場を提供する CS プラットフォームである．

CS システムには，選択演算や結合演算など，DB 操作に対応する処理がしばしば含まれる [4]．このような処理においては，例えば人手による特定主題の画像の選定（選択演算に対応）や，人手による画像の被写体の同一性判定（結合演算に対応）等，部分的に人手の作業が含まれる．CS システムにおけるこれらの演算（以下，**Human-powered 演算**）の処理については，近年活発に議論が行われている [4] [5] [6] [7]．

本稿では特に，データ統合等の場面において重要な演算である Human-powered Join の処理効率化について議論する．Human-powered Join とは，人手によって結合条件の判定を行う結合演算である．一般に，計算機による結合条件の判定が難しい場合に利用される．例えば，画像の被写体の同一性を判定する作業を人手によって行い，同一人物の顔写真のペアだけを選択するような演算である．

この演算に関する既存の議論では，次の 2 つの仮定を前提としている．（仮定 A）人手の作業は群衆の誰でも同様に行うことができる，（仮定 B）CS システムのプログラマ自身が演算対象に関する固有の知識（ドメイン知識）を持つ．しかし，これらは現実には必ずしも成立するとは限らない．例えば（仮定 A）に関しては，画像の被写体の同一性を判定する作業であれば誰でも同様に行うことができると考えられるが，サッカー選手や俳優等の画像とその出身地を結びつける作業は人によって分野に得意不得意がある．（仮定 B）に関しては，演算対象に関するドメイン知識を必ずしもプログラマが全て事前に把握しているとは限らない．

本稿では，既存の議論で前提とする仮定が成立しない場合の Human-powered Join の処理効率化手法を提案する．本提案手法の特徴は，処理効率化自体をクラウドソーシングによって行うことである．

なお，本稿で提案する処理効率化手法は MTurk 等の既存 CS プラットフォームでも実現可能である．しかし，本提案を考慮した CS プラットフォームがあれば，さらに直接的に実現可能である．本稿では，そのような CS プラットフォームについても説明を行う．

本稿の構成は次の通りである．2 節では関連研究について述べる．3 節では提案手法について説明する．4 節では，我々が利用する CS プラットフォームについて説明する．5 節では，提案手法を評価するための予備的な実験について説明する．6 節はまとめと今後の課題である．

2. 関連研究

近年，Human-powered 演算の処理について，多くの論文で議論されている．[4] [5] [6] [7].

論文 [4] では，CS プラットフォームである MTurk 上での人

図 1　Human-powered Join の作業例：画像の被写体の同一性判定

```
SELECT R.img, S.img
FROM R JOIN S
ON samePerson（R.img, S.img）
```

図 2　拡張 SQL [5] を用いた Human-powered Join の記述例

手の作業を含めたソート演算（Human-powered Sort）と結合演算（Human-powered Join）という 2 つの演算の処理効率化手法が提案されている．この効率化手法では，1 節で述べた仮定を前提としている

本論文では，既存の処理効率化手法と異なり，これらの仮定を前提としない手法を提案する．本提案手法の特徴は処理効率化自体もクラウドソーシングによって行う点にある．具体的には，処理効率化に関する次の 2 つのヒントをクラウドソーシングによって得る．（H_α）各作業を群衆のうちの誰が行うべきか，（H_β）作業をどのように分割し全体の作業量を削減するか．提案手法の詳細は 3 節で述べる．

3. 提案手法

本節では，まず Human-powered Join の例を示し，次にその効率化処理について例を用いて説明する．そして，この効率化処理のモデル化を行い，最後に提案手法を説明する．

3.1 Human-powered Join の例

Human-powered Join とは，人手によって結合条件の判定を行う結合演算である．例えば，図 1 のような画像の被写体の同一性を判定する作業を人手によって行い，同一人物の顔写真のペアだけを選択するといった結合演算が考えられる．（画像は Japanese Female Facial Expression (JAFFE) Database [8] を利用）．

この例において，それぞれが画像の集合を表す 2 つのリレーションを $R\,(img)$ と $S\,(img)$ とし，結合条件 p を「$R.img$ と $S.img$ の被写体が同一である」ことであるとする．ここで，論文 [5] で提案されている拡張 SQL を用いると，この Human-powered Join を図 2 のように記述することができる．この問合せが与えられると，R と S の全ての img のペアに対して，図 1 のようなフォームから「Yes（一致）」または「No（不一致）」を入力する．そして「Yes（一致）」が入力された img のペアが結合結果に含まれる．

3.2 Human-powered Join の効率化処理

図 2 の問合せを効率化する手法として，人手による作業を次

```
SELECT R.img, S.img
FROM R JOIN S
ON samePerson（R.img, S.img）
AND POSSIBLY gender(R.img) = gender(S.img)
```

図 3　論文 [4] が提案する図 2 の問合せの効率化処理の記述例

の 2 つに分割し，比較する画像のペアを少なくすることで全体の作業量を削減することが考えられる．

（1）　全ての画像に対して性別を入力

（2）　同性同士の画像のペアに対して同一性を判定

2. 節で述べた論文 [4] では，図 3 のような問合せを提案している．具体的には，4 行目に *Possibly* 句を追加することで，（1）に対応する処理を記述している

この問合せが与えられると次のように処理が行われる．まず，R と S に関連属性 `gender` を追加し，$R'(img, gender)$ と $S'(img, gender)$ を作成する．次に，R' と S' の全てのタプルに対して，人手によって `gender` 属性の値 `male` または `female` を入力する．そして，R と S の `gender` の値が一致するタプル同士を結合し，これらのペアに対して，人手によって「Yes（一致）」または「No（不一致）」を入力する．最後に「Yes（一致）」が入力された img のペアが結合結果に含まれる．

以上のような効率化を行うことで，例えば R と S に含まれる `img` の男女比が 1 : 1 である場合には，人手による作業の数は $|R \times S|$ から $\frac{|R \times S|}{2} + |R| + |S|$ に削減される．

3.3 Human-powered Join の効率化処理のモデル化

人手による作業を行う人の集合を表すリレーション $Crowd$ のスキーマを $Crowd(cid)$ とする（ここで cid は人の識別子である）．この時，リレーション R と S の条件 p による Human-powered Join の効率化処理は次のようにモデル化することができる．

（1）　R, S に関連属性群を追加した R', S' に属性値を入力

（2）　$T' = R' \bowtie_{equiv} S'$ を計算

（3）　$\pi_{attr(R \bowtie S)}(\sigma_{result=Yes}(\mathbf{J}_{cid,p,result}(T' \bowtie_{InCharge} Crowd)))$ を計算

以下でそれぞれのステップについて順に説明する．ステップ（3）については各演算ごとに分けて説明する．

（1）各タプルに，同値類に分割するための属性値を追加する．3. 2 節の例では，関連属性 `gender` を追加した R' と S' の作成および，人手による `male` または `female` の入力に対応している．

（2）$\bowtie_{equiv}$ は同値類（equivalence class）であるタプル同士を結合する結合演算である．3. 2 節の例では，関連属性 `gender` の値が一致するタプル同士を結合する処理に対応している．$\bowtie_{equiv}$ によって，ステップ（3）において異なる同値類に属するタプル同士の比較が行われないようになるため，人手による作業量の削減が達成される．

（3）本ステップで処理が行われる各演算について順に説明する．

- $T' \bowtie_{InCharge} Crowd : T'$ の各タプルに対して，作業を行う人 cid_i を選択する演算である．上記例では，`gender` の値が一致する各タプル $t \in T'$ に対してひとりずつ人を選択する．
- $\mathbf{J}_{cid,p,result}(e) : e$ の各タプルに対して，cid_i に対応する

人が結合条件 p に関する Yes/No を入力する演算である．入力された Yes/No は，新たな属性 $result$ の属性値として追加する．上記例では，結合条件 p「$R.img$ と $S.img$ の被写体が同一である」に関して「Yes（一致）」または「No（不一致）」を入力する．

- $\pi_{attr(R\bowtie S)}(e)$：通常の射影演算であるが，その射影属性は R と S に含まれる属性である．

以上の効率化処理のモデルを用いると，2 節で述べた既存の処理効率化における 2 つの仮定は次のように説明できる．(仮定 A) 人手の作業は群衆の誰でも同様に行うことができるという仮定は，$T' \bowtie_{InCharge} Crowd$ をシステムが機械的な処理で自動的に選択しても効率化処理が可能であることに対応する．(仮定 B) CS システムのプログラマ自身が演算対象に関するドメイン知識を持つという仮定は，同値類に分けるための関連属性群（先の例では gender）が，問合せの記述時に必ず指定可能であることに対応する．

3.4 提案手法

本提案手法の特徴は，処理効率化自体もクラウドソーシングによって行う点にある．具体的には，処理効率化に関する次の 2 つのヒントをクラウドソーシングによって得る．(H_α) 全体の作業量を削減するために，問題をどのように分割すべきか．(H_β) 各作業を群衆のうちの誰が行うべきか，

本稿では，これらのヒントを得るために，組み合わせて利用可能な独立した 2 つの手法を提案する．第一の手法（**A-Crowdsourcing**）では，同値類に分割するための関連属性群の追加をクラウドソーシングによって行う．追加された関連属性群は，ステップ（2）における $R' \bowtie_{equiv} S'$ の処理，およびステップ（3）における $T' \bowtie_{InCharge} Crowd$ の処理どちらにも利用される．第二の手法（**C-Crowdsourcing**）では，$T' \bowtie_{InCharge} Crowd$ の処理をクラウドソーシングによって行う．

以降では，これらの手法の説明を，次の例を利用して行う．

[問合せ Q1] 筑波大学の春日エリアの部屋番号（7D202 等）の一覧を表すリレーション R と，研究室名（森嶋研等）の一覧を表すリレーション S の結合

3.5 A-Crowdsourcing

A-Crowdsourcing では，R および S の関連属性群を追加した R', S' を作成する．ここでのチャレンジは，結合演算における問題の分割という高度な作業を，小さなタスクの集まりに分割する事によって実現する事である．

一般に，3.3 節のステップ（1）において関連属性群の個数は既知ではない．そこで，本提案手法では個別の属性（例えば gender）を追加せず，これらの属性の値の和集合を表す属性 A を追加する．また,「男であれば女でない」等の各関連属性に関する排他性の知識を明示的に扱うための属性 $\neg A$ を追加する．属性 A と属性 $\neg A$ の型はどちらも語の集合である．

A-Crowdsouring では関連属性 A と $\neg A$ を次の 4 つのステップのクラウドソーシングによって得る．

Step 1: 分割ラベル入力　仕事を担当する各 $c \in Crowd$ が，問題の分割に利用可能なラベルを表すキーワードを図 4 のようなフォームを通じて入力する．入力された一覧は常に参加者に見えるようにする．

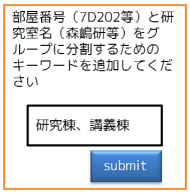

図 4　**Step 1** ：分割ラベル入力

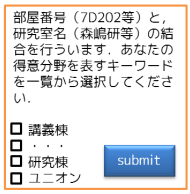

図 5　**Step 2** ：得意分野入力

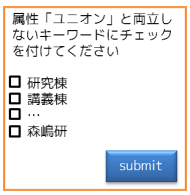

図 6　**Step 3**：非両立ラベルの発見

Step 2: 得意分野入力　各 $c \in Crowd$ が，得意分野を表す語の集合を図 5 のようなフォームを通じて入力する．入力された語の集合は $Crowd(cid)$ リレーションの新たな属性 $keywords$ の属性値として追加する．$keywords$ の型は語の集合である．入力された全ての語のを表すリレーションを $KEYWORDS = \{(w)|w \in t[keywords], t \in Crowd\}$ とする．例えば，図 8 では $c \in Crowd$ の得意分野を表す語として，$KEYWORDS =$ { 春日，講義棟，研究棟，ユニオン } が入力されている．

Step 3: 非両立ラベルの発見　全ての $w \in KEYWORDS$ に対して，全ての $w' \in KEYWORDS(w' \neq w)$ のうち，w と両立しない語（“男” と “女” 等）の集合に図 6 のようなフォームを通じてチェックを付ける．チェックのついた全ての w' は $KEYWORDS$ リレーションの新たな属性 $incompatible\_keywords$ の属性値として追加する．例えば，図 8 では「研究棟」と両立しない語の集合として { ユニオン } が入力されている．

Step 4: 属性値の追加　R' と S' の各タプルに対して A の属性値および $\neg A$ の属性値を図 7 のようなフォームを通じて入力させる．例えば，図 8 では R' の部屋番号が「7D130」である属性 A の値として { 春日 }，属性 $\neg A$ の値として { 森嶋研 } が入力されている．

以上のステップで，クラウドソーシングによって入力された属性 A と $\neg A$ は，3.3 節でモデル化した Human-powered Join の効率化処理のステップ（2）および（3）において次のように利用される．

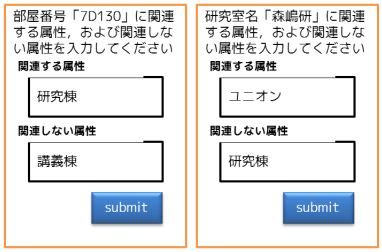

図 7　**Step 4**：属性値の追加

Crowd

cid	keywords
1	{春日，森嶋研}
2	{研究棟}
…	
3	{ユニオン}
4	{講義棟}

R′

部屋番号	A	¬A
7D130	{春日}	{森嶋研}
7D131	{}	{ユニオン}
7D140	{}	{ユニオン}
…		
7D541	{}	{ユニオン}
7D542	{春日}	{}

S′

研究室名	A	¬A
池内研	{研究棟}	{}
石井研	{}	{春日}
…		{}
村井研	{研究棟}	{}
森嶋研	{}	{ユニオン}

KEYWORDS

keywords	incompatible_keywords
春日	{}
森嶋研	{研究棟}
…	…
研究棟	{ユニオン}
ユニオン	{研究棟}

図 8　A-Crowdsouring で作成するリレーションの例

（2） $R' \bowtie_{equiv} S'$ を次の演算で実装する．

$$R' \bowtie_{\neg(R'.A \cap S'.\neg A \neq \phi \vee R'.\neg A \cap S'.A \neq \phi)} S'$$

すなわち，R' の属性 A の値が，S' の属性 $\neg A$ に含まれる場合，タプルの結合を行わない．S' に対しても同様である．例えば，図 8 では R' の部屋番号が「7D130」である属性 A の値として { 春日 }，S' の研究室名が「石井研」である属性 $\neg A$ の値として { 春日 } が入力されている．したがって，これらのタプルの結合は行わない．

（3） $T' \bowtie_{InCharge} Crowd$ を次の演算で実装する．

$$T' \bowtie_{\neg(Crowd.keywords \cap R'.\neg A \neq \phi \vee Crowd.keywords \cap S'.\neg A \neq \phi)} Crowd$$

すなわち，R' もしくは S' の属性 $\neg A$ の値が，$Crowd$ の属性 $keywords$ に含まれる場合，タプルの結合を行わない．例えば，図 8 では $Crowd$ の cid が「3」である属性 $keywords$ の値が { ユニオン } であり，R' の部屋番号が「7D131」である属性 $\neg A$ の値も { ユニオン } である．したがって，$T' = R' \bowtie_{equiv} S'$ のうち部屋番号が「7D131」であるいかなるタプルとも，cid が「3」であるタプルの結合は行わない．

3.6　C-Crowdsourcing

C-Crowdsourcing では，$\bowtie_{InCharge}$ をクラウドソーシングによって行う．具体的には，$\sigma_{result}(\mathbf{J}_{cid,p,result}(T' \bowtie_{InCharge} Crowd))$ の処理の過程で，次のように動的に cid を書き換え $J_{cid,p,result}$ の再計算を行う．

（1）　$\mathbf{J}_{cid,p,result}(T' \bowtie_{InCharge} Crowd)$ において，結合条件 p を判定するタプル t への入力として Yes/No だけでなく，「Unknown」もしくは「Yes/No の入力が可能と考えられる人に対応する cid_i」を入力できるようにする

（2）　t の cid 属性の値を次のように書き換え，t に関する結合を再計算する

- 「Unknown」が入力された場合，ランダムに決定した $cid_j (j \neq i)$
- 「cid_i」が入力された場合，cid_i

以上の単純な C-Crowdsourcing では，再割り当てを行う対象が $Crowd$ に登録されている人に限定されている．しかし，それ以外の人に割り当てるように拡張することが可能である．これは，選択肢として，「Yes」「No」「Unknown」「cid」の他に「Yes/No を知っていると考えられる人の連絡先」を入力することによって実現する．本稿では，具体的な連絡先として，メールアドレスとマイクロブログのアカウント名を扱う．連絡先が入力された場合，システムはその人に対して Yes/No の入力を求めるフォームを通知する．本稿では，この手法を**拡張 C-Crowdsourcing** と呼ぶ．

3.7　効率化に関するその他の議論

以上の A-Crowdsourcing と C-Crowdsourcing 手法は独立しているため，組み合わせて利用することができる．すなわち，A-Crowdsourcing の処理を行い，かつ $\sigma_{result}(\mathbf{J}_{cid,p,result}(T' \bowtie_{InCharge} Crowd))$ の処理の過程で，次のように動的に cid を書き換え $\mathbf{J}_{cid,p,result}$ の再計算を行うことができる．本稿ではこの処理を **AC-Crowdsourcing** と呼ぶ．

また，本稿では主要な論点ではないが，提案する 2 つのクラウドソーシングによる手法とは独立した効率化手法も利用可能である．例えば，結合対象となるスキーマでの関数従属性を利用して，Human-powered Join における人手による作業の数を削減することができる．3.4 節の例において「研究室名 → 部屋番号」という関数従属性が存在する場合には，研究室に対応する部屋番号がひとつ確定した後は，その研究室名と別の部屋番号が対応するかどうか判定する必要はない．

4.　提案プラットフォーム

本稿における C-Crowdsourcing では，クラウドソーシングによる作業者が，他の作業者とコミュニケーションを取ることが必要となる．これは既存 CS プラットフォーム（MTurk 等）でも実現可能であるが，このようなコミュニケーションを支援する CS プラットフォームがあれば，より直接的に実現可能である．

我々は，そのような CS プラットフォーム Crowd4U の開発を行っている．Crowd4U の特徴は次の 2 つである．（1）作業を行う人が，割り当てられた作業を他の作業者やプラットフォームに登録されていない知り合いに依頼するための機能を持つ．（2）既存の CS プラットフォームである MTurk 等とは異なり，金銭以外のインセンティブによるクラウドソーシングの実現を目指している．

5. 予備実験

本節では，提案する Human-powered Join の効率化手法に関する予備的な実験について説明する．

5.1 実験目的

実験の目的は，本稿で提案する **A-Crowdsourcing** と **C-Crowdsourcing** という 2 つの手法の可能性や手法の適切性などについて議論するためのデータを得る事である．

5.2 実験方法

本予備実験では，まず Human-powered Join を行う問合せをひとつ用意する．次に，同じ問合せに対して，効率化手法の組み合わせたシミュレーションを行い，クラウドソーシングによる問題の分割と，人のリクルーティングの効果に関して考察を行う．

予備実験で用意する問合せとしては，3.4 節で述べた問合せ Q1 を用いた．問合せ Q1 を選んだ理由は，既存の処理効率化における仮定が成立しないと考えられるからである．第一に，研究室名と部屋番号を結合することは，人によって得意不得意があり，誰でも同様に行えることではない．第二に，問合せ Q1 の効率化のための知識は，人物のマッチングの際に男女に分けるというものと比較して，誰もが持っている訳ではない．対象とするデータは，筑波大学春日エリアの研究室部屋番号一覧を表すリレーション R（タプル数 53）と研究室名を表すリレーション S（タプル数 64）である．

本予備実験においては，次の 2 つの実行過程をシミュレーションし，その結果に基づき，クラウドソーシングによる問題の分割と人のリクルーティングの効果を議論する．

（A）完全なドメイン知識を用いた分割． 比較対象として，クラウドソーシングを用いず完全なドメイン知識を持つ人が理想の問題分割を行ったと仮定してシミュレーションを行う．具体的には，分割リレーション R と S にそれぞれ問題分割のための追加属性「建物名」を追加した場合（分割 1）と，「建物名と階数」を追加した場合（分割 2）を設定する．また，全ての教室と教員の建物名と階数は全て既知であると仮定する．

（B）拡張 AC-Crowdsourcing． 提案した A-Crowdsourcing と拡張 C-Crowdsourcing を組み合わせたと仮定してシミュレーションを行う．*Crowd* としては 4 名の被験者を用意する．これら 4 名は，全体として問合せ Q1 に関する知識をもっているものの，各自は部分的な知識しか持っていない人を組み合わせて構成されている．この *Crowd* を利用して具体的には次を行う．まず，A-Crowdsourcing の手法をシミュレートし，同値類への分割を行う．次に，分割された同値類に対して，$T' \bowtie_{inCharge} Crowd$ の処理を行う．担当できないタスクに対しては，C-Crowdsourcing のシミュレーションで協力者を募り，この経過を記録する．今回は，ラベルを用いた割り当てに焦点を当てて調査を行うため，共に追加属性値が付与されているタプルの組み合わせに対してのみ C-Crowdsourcing のシミュレーションを行った．

5.3 実験結果と考察

クラウドソーシングによる問題分割． 図 9 は，問題分割を行わない場合，分割 1 および 2，A-Crowdsourcing のシミュレーション（A-CS）による最終的なタスク数を比較したものである．問題分割を行わない場合（分割無し），3392 のタスクが必要になるのに対し，分割 1 では 2230, 分割 2 では 1094 のタスクとなる．A-Crowdsourcing のシミュレーションでは，2148 のタスクが生成された．これは，理想的な分割 1 と 2 の間にあり，ほぼ分割 1 と同じ値である．したがって，*Crowd* が全体として対象ドメインの知識を持っているときには，それほど悪くない問題分割ができる可能性があることが示されている．

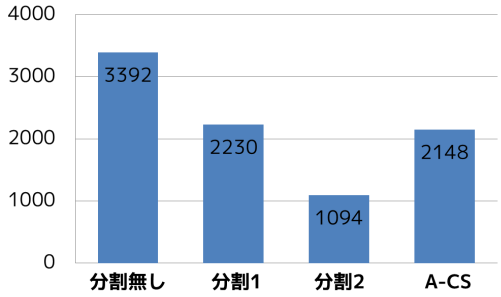

図 9　各手法による最終的なタスク数

ドメイン知識の仮定． A-Crowdsourcing の結果として生成された 2148 のタスクのうち，二つのタプルのどちらにも追加属性値が付与されていたタスクはわずか 187 であった．今回は，ある程度ドメイン知識を持つ人によって作業が行われたにもかかわらずこのような結果になったということは「誰でも同じ作業を行うことができる」という仮定によるクラウドソーシングの限界を示していると言える．したがって，これを仮定しないアプローチの重要性が再確認できたと考えられる．

クラウドソーシングによる人のリクルーティング． 二つのタプルのどちらにも追加属性値が付与されていた 187 のタスクを対象に，*Crowd* によるタスク遂行可能率に関する C-Crowdsourcing の効果を比較した．タスク遂行可能率とは，タスク集合が，*Crowd* に含まれる人によって遂行可能である割合である．実験の結果，初期のタスク遂行可能率は約 87.2%であり，C-Crowdsourcing のシミュレーションで協力者を募り，4 名をリクルートした後のタスクの遂行可能率は 100.0%となった．この結果から，クラウドソーシングによる人のリクルーティングの効果は期待できると考えられる．

6. まとめと今後の課題

本稿は，クラウドソーシングによる Human-powered Join の効率化手法について提案を行った．既存の処理効率化手法に対して，本提案手法の特徴は処理効率化自体もクラウドソーシングによって行う点にある．

本稿では，独立した次の 2 つの効率化手法を提案した．（1）全体の作業量を削減するために作業をどのように分割すべきかというヒントをクラウドソーシングによって得る手法（A-Croudsourcing），（2）各作業を群衆のうちの誰が行うべきかというヒントをクラウドソーシングによって得る手法（C-Crowdsourcing）．

今後の課題としては，より多様なシナリオを用いた詳細な実験による知見の入手とその結果に基づく提案手法の改良などが挙げられる．また，クラウドソーシングを行う作業の単位を変えた場合の提案化手法の有効性の検討等も今後の課題である．

謝　　辞

ゼミなどにおいてコメントいただきました，筑波大学 杉本重雄教授，阪口哲男准教授，永森光晴講師に感謝いたします．また評価実験に際してご協力いただきました研究室の後輩の宮田愛さんに感謝いたします．本研究の一部は JST さきがけ「情報環境と人」および科学研究費補助金（#21240005）による．

文　　献

[1] gwap.com. “ESP Game”. http://www.gwap.com/gwap/gamesPreview/espgame/
[2] reCAPTCHA. “What is reCAPTCHA?”. http://recaptcha.net/learnmore.html
[3] Doan AnHai, Ramakrishnan Raghu, Halevy Alon Y.. “Crowdsourcing systems on the World-Wide Web. Commun.ACM. 2011, vol. 54, no. 4, p. 86-96.
[4] Marcus Adam, Wu Eugene, Karger David R., Madden Samuel, Miller Robert C.. “Human-powered Sorts and Joins”. PVLDB. 2011, vol. 5, no. 1, p. 13-24.
[5] Marcus Adam, Wu Eugene, Madden Samuel, Miller Robert C.. “Crowdsourced Databases: Query Processing with People”. CIDR. 2011, p. 211-214.
[6] Franklin Michael J., Kossmann Donald, Kraska Tim, Ramesh Sukriti, Xin Reynold.. “CrowdDB: answering queries with crowdsourcing”. SIGMOD Conference. 2011, p. 61-72.
[7] Parameswaran Aditya G., Polyzotis Neoklis.. “Answering Queries using Humans, Algorithms and Databases”. CIDR. 2011, p. 160-166.
[8] Michael J. Lyons, Akamatsu Shigeru, Kamachi Miyuki, Gyoba Jiro. “Coding Facial Expressions with Gabor Wavelets”. Third IEEE International Conference on Automatic Face and Gesture Recognition. 1998, p. 200-205.