DBSJ Japanese Journal
Vol. 14-J, Article No. 8, March 2016

クラウドソーシングデータ処理における一般化選択フィルタの利用と動的選択手法

A Method for Dynamic Selection of Generalized Selection Filters in Crowdsourced Data Processing

権守 健嗣♡ 森嶋 厚行◇ 歳森 敦♠ 北川 博之♣

Kenji GONNOKAMI Atsuyuki MORISHIMA Atsushi TOSHIMORI Hiroyuki KITAGAWA

クラウドソーシングシステムにおける重要な課題の一つとして最適化がある．クラウドソーシングシステムの多くは，ヒューリスティクスを用いた最適化を取り入れているが，どのヒューリスティクスが有効であるかは必ずしも自明でないため，プロトタイピングなどによるシステム開発者の手間の増大や，ヒューリスティクス選別のために必要なタスク数が増大するなどの問題が生じる．本論文では，本番処理中に動的にヒューリスティクスの選別を行う事により，ヒューリスティクス選別のためのプロトタイピングを必要とせず，タスク数も増大しない手法を提案する．提案手法は，ヒューリスティクスを一般化選択フィルタとしてモデル化し，有効なフィルタを動的に選別する．本論文では，提案手法の説明に加え，実データを用いたシミュレーションの結果を示す．

One of the important issues in crowdsourcing is optimization. Many crowdsourcing applications adopt heuristic-based optimization, but choosing effective heuristics for the application is a cumbersome task. This paper proposes a method that chooses effective heuristics during the crowdsourcing process, which makes a preprocessing to choose the heuristics unnecessary. The proposed method models each heuristic as a generalized selection filter and chooses effective ones during the crowdsourcing process. This paper explains the method and shows the results of our simulation with a set of real-world data.

1. はじめに

計算機ネットワーク技術の発達に伴い，不特定多数の群衆に仕事を委託するクラウドソーシングが多くの分野で用いられている．一般に，クラウドソーシングを行うシステムはクラウドソーシングシステムと呼ばれている [3]．本論文では，短時間で処理可能なタスク（マイクロタスク）を用いたクラウドソーシングシステムを対象とした最適化について議論する．

クラウドソーシングシステムでは，処理が長期にわたることが少なくないため，最適化がより重要な課題となっている．クラウドソーシングシステムの最適化においては，人手によるタスク処理コストが非常に高いため，少しでも人手によるタスク数を削除することが重要となる．したがって，常に成立する変換規則を用いた最適化だけでなく，対象データに関してのみ成立するデータのセマンティクスに依存した最適化や，ヒューリスティクスを用いた最適化がしばしば行われる．例えば，同一人物の写真かどうかを判定するために，同じ髪の色であるかどうかであらかじめフィルタリングするという事例 [5] がある．

ヒューリスティクスは必ずしも正しいとは限らないが，クラウドソーシング処理においては役に立つ場合がある．例えば，クラウドソーシングシステム処理においては，実行時間が長期にわたるため，中間結果の再現率が高いことが望ましい場合がままある．具体的には，被災地の航空写真の集合から火事の特定をしたいような場合には，画像処理などで赤みがかった画像を発見し，それらを優先して探すことにより，そのような箇所をより早く発見できる可能性がある．

しかし，どのヒューリスティクスが効果的であるかどうかはデータセットの性質に依存する．例えば，果物の写真の集合からリンゴの写真を選択したいとき，「写真に赤い物体が認識された場合，写真にリンゴが写っている可能性が高い」というヒューリスティクスを考える．しかし，このヒューリスティクスは，写っている果物の全てが赤みがかっていた場合には，有効ではない．もし，そのような写真の集合からりんごを探す場合には，丸い果物が写っているというヒューリスティクスのほうが適切である．もちろん，梨など丸い果物が多く写った写真の集合から探す場合には先に述べた赤い物体が認識された場合というヒューリスティクスのほうが適切である．このように，どのようなヒューリスティクスを適用すべきかはデータセットの性質，セマンティクスによって判断する必要がある．

しかし，クラウドソーシングを行うような状況では，あらかじめデータセットの性質がわからないという場合も多い．このような場合，次のような 2 フェーズのアプローチを取ることが典型的である．まず，有効なヒューリスティクスの選別を行うためにタスク処理の試行を繰り返すフェーズを行い，次に，選別を終えたヒューリスティクスを用いて本番のタスクを行うフェーズを行う．この 2 フェーズによるクラウドソーシングは，二つの問題がある．第一に，タスクを作成する人（リクエスタ）が行う作業負担が大きいために，本番のタスクの開始までに時間がかかることである．第二に，タスク数の増大である．なぜなら，第一フェーズでワーカにタスクを行ってもらう場合には本番のタスクとは別のタスクが必要になるからである．

本論文では，リクエスタが適用したいヒューリスティクスを全て入力すると，フェーズ 1 を行うことなく，システムの実行中に実際のデータセットに応じてヒューリスティクスを動的に切り替える枠組みを提案する．

本論文では，クラウドソーシングのヒューリスティクスを，データベース処理の最適化でしばしば利用される選択フィルタ (Selection Filter) を一般化した概念である**一般化選択フィルタ** (Generalized Selection Filter) としてモデル化する．選択フィルタとは，選択演算 σ_p の処理コストが高いときに，事前に演算対象のデータサイズを削減する選択演算 σ_q の事である．その結果，選択のトータルコストを削減する．例えば，リレーション R のタプル中から，インデックスが張られていない文字列属性にニホンオオカミという単語が含まれているタプルを選択する演算 $\sigma_p(R)$ を考える．このとき，その選択フィルタとして，文字列の言語情報が Japanese であるタプルだけを選択する演算 $\sigma_q(R)$ を利用する場合，その式は $\sigma_p(\sigma_q(R))$ と書ける．本論文では，$\sigma_q(R)$ が条件 p の選択演算

♡ 非会員　筑波大学大学院システム情報工学研究科
kenji.gonnokami.2012b@mlab.info
◇ 正会員　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
mori@slis.tsukuba.ac.jp
♠ 非会員　筑波大学図書館情報メディア系
tosimori@slis.tsukuba.ac.jp
♣ 正会員　筑波大学システム情報系
kitagawa@cs.tsukuba.ac.jp

recall\precision	< 1	=1
< 1	Partial Coarse Filter	Partial Filter
=1	Coarse Filter	Perfect Filter

図 1: 一般化選択フィルタの分類
Figure 1: Classification of generalized selection filters

のための選択フィルタ演算であることを強調するため，$\sigma^p_q(R)$ と書く．

これまで，様々な選択フィルタが提案されている．例えば，Bloom Filter[1] は有名な選択フィルタである．また，Semantic Optimization ではデータの性質に依存した選択フィルタが利用される [2]．選択フィルタの共通の特徴は，false positive を含むが，false negative を含まないという事である．すなわち，正解集合 $\sigma_p(R)$ に対して $\sigma^p_q(R)$ の再現率と適合率を計算すると，再現率は 1 である事が保証されている．

一般化選択フィルタは，この制約を除去したものであり，4 種類に分類できる (図 1)．図中の Coarse filter および Perfect Filter が，通常の選択フィルタに対応する．ヒューリスティクスを利用した選択フィルタは，一般には Partial Coarse Filter になる．例えば，p を火災の写真，q を赤みがかった写真とすると，σ^p_q は Partial Coarse Filter であると考えられる．

本論文の貢献は次の通りである．

(1) 一般化選択フィルタによるクラウドソーシングヒューリスティクスのモデル化． クラウドソーシングで利用されるヒューリスティクスを一般化選択フィルタとしてモデル化する事により，全体の問合せ処理の中での最適化の議論などが行いやすくなる．q であれば p である (もしくは可能性が高い) というヒューリスティクスを利用した選択フィルタは $\sigma^p_q(R)$ と表現できる．また，一般化選択フィルタは，機械学習などを用いて得られた Classifier もモデル化可能である．すなわち，x を p と $\neg p$ に分類する分類器 $C(x)$ を使った一般化選択フィルタは，$\sigma^p_{C(x)=p}(R)$ と表現できる．このように，一般化選択フィルタは，クラウドソーシング最適化で利用できる様々な手法を統一的にモデル化可能である．

(2) 選択フィルタの動的な選別． 本論文では，データセットに応じて，フィルタを選別する手法を提案する．一般化選択フィルタの適合率はデータセットの性質に依存するため，効果的な一般化選択フィルタはデータセットによって異なる．そのため，データセットに応じて適切な一般化選択フィルタを選ぶことが望ましい．本手法では，本番のタスクの処理の途中結果を元にフィルタを選択することで動的な選択を可能とする．

(3) 2 フェーズでない選択フィルタの選別． 本論文では，2 フェーズの処理を行うことなく，効率的にフィルタを選別する手法を提案する．本手法のアイデアは次の通りである．あるクラウドソーシングシステムにおいて，効果的な可能性があるヒューリスティクス (一般化選択フィルタ) 集合を $F_{ALL} = \{\sigma^p_{q1}, \sigma^p_{q2}, ...\}$ とする．リクエスタは，提案手法の入力として F_{ALL} を全て登録する．提案手法は，効果的である可能性があるフィルタ集合 F の初期値を登録された F_{ALL} とする．そして，本番のタスクの処理結果を集めながら，F 中のフィルタの適合率に有意差があるかどうかを処理結果の集計を元に検定し，有意に劣っていると判断できたフィルタから順に，F から取り除いていくことになる．そのため，フィルタの選別のためのタスクと本番のタスクを区別する必要がない．

(4) 他の結果に依存する一般化選択フィルタの適用． 本手法では他の結果に依存する一般化選択フィルタを扱える．クラウドソーシングシステムでは，このようなヒューリスティクスを扱える必要がままある．例えば，竜巻によって壊れた建物を探す問題において，壊れている建物の近くは壊れているというヒューリスティクスは効果的であると考えられるが，各タスクにおいて近くにある建物が壊れているかどうかは他のタスクを処理するまでわからない．本手法では，このようなヒューリスティクスを表す一般化選択フィルタも選別の対象に加えられる．具体的には，一般化選択フィルタの性能に有意に差があるとわかるまで，選別の判定を保留することで可能とする．

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 節では一般化選択フィルタのモデル化，一般化選択フィルタの応用について説明する．4 節では提案手法におけるフィルタの選別について説明する．5 節では，提案手法の有効性を確かめるためのシミュレーションについて述べる．6 節はまとめと今後の課題である．

2. 関連研究・関連システム

Semantic Query Optimization (SQO) 本手法に関連する研究として SQO[2] が挙げられる．SQO は一般には等価でないクエリ間をドメイン知識を考慮することで変換し，最適化するものである．SQO は確実なドメイン知識から等価なクエリを得るものであるが，本手法は，不確実なドメイン知識を元に考えられるヒューリスティクスからタスクの優先度を与えるものである．一般に，クラウドソーシングでは確実なドメイン知識を与えるのは難しいため，本手法は有益と考えられる．

Multiple Classifier System (MCS) 本手法におけるフィルタの選別と関連する研究として，MCS[9] が挙げられる．MCS に関する研究は，複数の Classifier を用いて，よりよい分類結果を得るためのものである．MCS に関する研究は複数のアプローチで行われているが，本手法に最も関連するアプローチとして Classifier Selection が挙げられる．Classifier Selection は複数の Classifier から，優れている Classifier を選ぶ問題であるのに対し，本手法は複数の一般化選択フィルタから，最も優れたフィルタを選ぶ点で似ている．

特に，与えられたデータセットに対して正しそうな Classifier を予測して選択するような Classifier Selection は Dynamic Classifier Selection と呼ばれている．Dynamic Classifier Selection の研究としては，[10][4] がある．[10] はトレーニング済みのデータから分類するデータに近いものをいくつか集め，それに対する正確度を用いて，Classifier を選別する手法を提案している．[4] は，Dynamic Classifier Selection を行い，最適なベイズ分類器を選ぶ手法について研究したものである．

Dynamic Classifier Selection は全体の候補からデータセットに合わせて適したものを選ぶという点で本研究と共通する．しかし，次のような異なる点がある．まず，フィルタの選別と Classifier Selection では，選別の尺度が異なる．Classifier Selection が選別の尺度として主に正確度 (Accuracy) を用いるのに対し，フィルタの選別では適合率 (Precision) を用いる．そのため，フィルタの選別には正例のみを集めればよい．次に，Dynamic Classifier Selection は Classifier の選別と適用の 2 フェーズであるのに対し，本手法は 1 フェーズで行うものである．また，人的リソースの制限という点でも異なる．一般に，Classifier Selection では限られたサンプルデータ（人手で作られた正解データ）を用いて性能の推定を行うのに対し，本研究では，タスクは最終的に全て行われることが前提であるため，フィルタの選別に用いる人的リソースに制限はない．

また，本手法では他の結果に依存するようなフィルタを扱うことができるので，Dynamic Classifier Selection における前提の一つである「分類するデータの特徴量は既知である」を満たす必要がない．加えて，Dynamic Classifier Selection は対象とするデータセットの各データの特徴ベクトルを明示的に与える必要があるが，本手法では特徴ベクトルを明示的に扱わない．これらの前提をクラウドソーシングにおいて満たすのは難しいため，クラウドソーシングに一般化選択フィルタを応用する上で，これらの前提を必要としないことは重要である．

MCS に関する研究の別のアプローチとして Classifier Ensemble がある．Classifier Ensemble は複数の Classifier を組み合わせることで優れた Classifier を生成する研究である．本論文で

は，Classifier Ensemble のように複数の候補を組み合わせることについては扱わないが，発展として複数のフィルタを組み合わせたフィルタを F に加えるということが考えられる．その場合の問題点は多数の組み合わせから，F に加えるべき組み合わせをいかにして発見するかということである．

3. 一般化選択フィルタ

本節では，一般化選択フィルタと，クラウドソーシングにおけるその応用について議論する．

3.1 一般化選択フィルタ

定義 1 (一般化選択フィルタ) リレーション R, 述語 p, q が与えられ，R 上で q が成立すると p が成立する事がある程度期待される時，これらに基づく選択フィルタを σ^p_q と記述し，それを用いた選択演算を次のように定義する．

$$\sigma^p_q(R) \equiv \sigma_q(R)$$

定義 2 (空フィルタ) $\sigma^p_\phi(R) \equiv \sigma_{True}(R)$ となるような σ^p_ϕ を空フィルタと呼ぶ．これは，任意の R に対して，$\sigma^p_\phi(R) = R$ となる選択フィルタである．

3.2 ヒューリスティクスに基づく一般化選択フィルタ

選択演算のためのヒューリスティクスを $q \Rightarrow p$ (p, q は述語) と記述する．例えば,「赤みがかった画像には，リンゴが写っている (ことがある程度期待される)」というヒューリスティクスは,「画像に赤みがかかっている $\Rightarrow$ その画像にリンゴが写っている」と表現する．このヒューリスティクスに基づく一般化選択フィルタは，$\sigma^p_q(R)$ として記述できる．

一般化選択フィルタは，ヒューリスティクス以外に基づいても作成可能である．例えば，ヒューリスティクスではないが *Recall* が 1.0 とはならない判断基準 (例えば，q が成立すれば 0.5 の確率で p が成立することが保証されている) に基づくフィルタが考えられる．また，x を p と $\neg p$ に分類する分類器 $C(x)$ に基づくフィルタ $\sigma^p_{C(x)=p}(R)$ もあり得る．

3.3 一般化選択フィルタの性能

定義 3 (一般化選択フィルタの性能) R に対する一般化選択フィルタ σ^p_q の性能を測る尺度として，再現率 *Recall* と適合率 *Precision* を次のように定義する．

$$Recall(\sigma^p_q, R) \equiv \frac{|\sigma_p(R) \cap \sigma^p_q(R)|}{|\sigma_p(R)|},$$

$$Precision(\sigma^p_q, R) \equiv \frac{|\sigma_p(R) \cap \sigma^p_q(R)|}{|\sigma^p_q(R)|}.$$

定義 4 (Coarse Filter)
$Recall(\sigma^p_q, R) = 1.0, Precision(\sigma^p_q, R) < 1.0$ であるような一般化選択フィルタを R に関する p の ***Coarse Filter*** と呼ぶ．

定義 5 (Partial Filter)
$Recall(\sigma^p_q, R) < 1.0, Precision(\sigma^p_q, R) = 1.0$ であるような一般化選択フィルタを R に関する p の ***Partial Filter*** と呼ぶ．

定義 6 (Partial Coarse Filter)
$Recall(\sigma^p_q, R) < 1.0, Precision(\sigma^p_q, R) < 1.0$ であるような一般化選択フィルタを R に関する p の ***Partial Coarse Filter*** と呼ぶ．

定義 7 (Perfect Filter)
$Recall(\sigma^p_q, R) = 1.0, Precision(\sigma^p_q, R) = 1.0$ であるような一般化選択フィルタを R に関する p の ***Perfect Filter*** と呼ぶ．

3.4 クラウドソーシングにおける一般化選択フィルタの応用

(1) よりよい中間結果の入手. クラウドソーシングにおいては，処理に時間がかかる場合が多いため，中間結果が重要である場合がしばしばある．例えば，災害時に写真を順に調査し，出来るだけ早く被災家屋を発見したい場合などでは，数少ないタスクで，再現率が高いような順番で，タスクを行う事が重要となる．そのような場合に，一般化選択フィルタは有効である．

それは，次の式が成立することを利用する．

$\sigma_p(R) =^{①} \sigma_p(\sigma^p_q(R)) \cup^{②} \sigma_p(R - \sigma^p_q(R))$

すなわち，選択フィルタを通ったタプルを先に処理しても後に処理しても，フィルタを使わなかった場合と結果が同じになるということである．したがって，①, ②の順序で計算することで，途中結果の再現率を高くできる可能性が高い．ただし，$Precision(\sigma^p_q, R) \leq Precision(\sigma^p_\phi, R)$ の場合には，中間結果の再現率は高くならない．

(2) 入力デバイスの切り替え. クラウドソーシングのマイクロタスクは，様々な環境で行われる．例えば，PC 上の画面で行われる場合もあれば，タブレットやスマートフォン，床やスクリーンロックで行われることもある [6][7][8]．これらにおいては，それぞれ適切な入力 UI が異なる．例えば，PC では複雑な入力が可能であるのに対して，タブレット等では選択式が望ましく，床などでは Yes/No 程度の選択肢に限定される．

一般選択フィルタはこれらをダイナミックに切り替えるのにも応用可能である．例えば，あるフィルタ適用後のタスク結果の度数分布において，最頻値が占める割合が 0.5 以上であるような場合に，その割合が有意に 0.5 以上であることがわかった時点で UI を切り替え,「最頻値」と「最頻値以外」という 2 択のタスクに切り替えられる．その場合，床やスクリーンロックなどの環境にタスクを割り当てる事が可能となる．

3.5 一般化選択フィルタと全体の計算量との関係

一般化選択フィルタでは false negative を含み得るため，最終的には，本来の選択演算を行う必要がある．そのため，一般化選択フィルタを用いたクラウドソーシング処理は，単純な計算量だけを見た場合には本来の選択演算のみに比べ大きくなる．

しかし，一般に長期にわたるクラウドソーシング処理では，ある程度小さな計算量での途中結果の改善を優先することがある．また，多くの場合，人が処理する p の計算量は，計算機が処理する q の計算量に比べ非常に大きいため，一般化選択フィルタの計算量が問題となることは稀であると考える．

また，もし，q の計算量が p の計算量に対して無視できないほど大きい場合であっても，本手法は，q の計算が完了するまで p の計算を制限するものではないため，実経過時間が本来の選択演算のみにかかるものよりも大きくなることはない．

しかし，上の議論は q の計算が計算機によって行われる場合に限られる．もし，q の計算を人が処理するような一般化選択フィルタを考えた場合には，本来の選択演算にかけるリソースを q の計算に割くことになるため，実経過時間も増大すると考えられる．そのような場合には，q の計算リソースを p の計算リソースとは別に確保するといったことが考えられる．例えば，$\sigma^p_q(R)$ において R を昆虫の写真の集合，p を蝶の写真，q を蝶もしくは蛾の写真とすると，蝶と蛾の区別が付く人物は限られるため，p の計算リソースは限られるが，他の昆虫と蝶もしくは蛾との区別が付く人物は比較的多いと予想されるので，q の計算は p の計算を行えない人に割り振ることが考えられる．その場合には，本来の計算リソースを損なうことにはならない．

本件に関しては，5.3 節で詳細に議論する．

4. 一般化選択フィルタの選別

本節では，提案する一般化選択フィルタの選別手法を組み込んだタスク処理アルゴリズムについて説明する．ここでのフィルタの

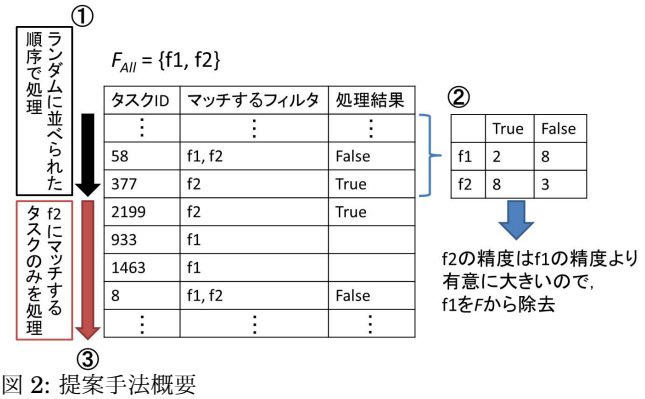

図 2: 提案手法概要
Figure 2: Overview of the proposed method

選別とは，リクエスタが登録した効果的である可能性のあるヒューリスティクス (一般化選択フィルタ) の集合 F_{ALL} から，適合率の優れたフィルタを選び，適合率の高いフィルタから順次適用することである．

4.1 フィルタの選別の概要

問題の簡単化のために，与えられたデータセットについての各フィルタの適合率が既知な場合を考える．この場合，フィルタの選別は適合率の高い順にフィルタを選ぶだけでよい．具体的には，次のようにする．まず，現時点で有効と考えられるフィルタの集合 F の初期値を F_{ALL} とする．次に，F から最も適合率の高いフィルタを取り出し，それにマッチするタスクを全て行い，マッチするタスクが無くなり次第，F から次に適合率の高いフィルタを取り出し，マッチするタスク全てを行う．このように，F から最も適合率の高いフィルタを取り出し，タスクを行うという手順を繰り返す方法が単純かつ効果的である．

提案手法は上で述べた手順に似ているが，実際には，各フィルタの適合率はわからないため，上の手順をそのまま実行することはできない．提案手法の概要を図 2 に，大まかなアルゴリズムを Algorithm 1 に示す．

本手法の入力は，本番のタスク集合 T と，効果的である可能性のある一般化選択フィルタの集合 F_{ALL} であり，出力は本番のタスクそれぞれの処理結果の配列（Algorithm 1 の results）である．

まずは，フィルタの性能を測るためのサンプリングを兼ねて，図 2 の ① に示すように，ランダムに並べられた順序でタスクを行う（Algorithm 1 の 2-7 行目）．次に，図 2 の ② のように，サンプリングの結果，有意に劣っていると判断されたフィルタ（図 2 では f1）を F_{pool} に移すことによって，消去法の形で F を選別する（Algorithm 1 の 8-11 行目）．そして，図 2 の ③ のように選ばれたフィルタにマッチするタスクだけを全て行う．マッチするタスクが無くなり次第，F を F_{pool}（図 2 では f1）で新たに初期化した上で，ランダムな処理に戻り，F を選別する（Algorithm 1 の 14 行目）．この手順を繰り返すことで，適合率の高い順にフィルタを適用するという元になった手順にできるだけ沿うようにする．

4.2 フィルタの比較

本手法では，二つのフィルタの適合率に有意差があるかどうか判定するため，検定を行う．具体的には，「$Precision(\sigma^{p}_{q1}, R) = Precision(\sigma^{p}_{q2}, R)$」という帰無仮説を検定する．

その方法としてフィッシャーの正確確率検定とカイ二乗検定を用いる．フィッシャーの正確確率検定は標本数が少ない場合にも正確な検定が行えるが，標本数が大きくなると計算量が膨大になることで知られている．そのため，本手法では，標本数が小さいときにはフィッシャーの正確確率検定を用い，標本数が大きくなったときにはカイ二乗検定を用いる．

Algorithm 1 提案手法の大まかなアルゴリズム
Simplified algorithm of the proposed method

Require: T, F_{ALL}
Ensure: results[0, ... , |T| − 1]
1: $F \leftarrow F_{ALL} \cup \{\sigma_{\phi}\}$
2: tasks ← order_randomly(T)
3: **while** ∃i (0 ≤ i < |T|) *s.t.* results[i] = NULL **do**
4: $F_{pool} \leftarrow \phi$
5: **for** i in [0 ... tasks.length - 1] **do**
6: **if** ∃$f \in F$ that matches $tasks[i]$ **then**
7: results($tasks[i]$) ← ask($tasks[i]$)
8: compare results($tasks[i]$) with ∀$f \in F$ that matches $tasks[i]$
9: **for all** $f \in F$ *s.t.* f は有意に劣る **do**
10: $F \leftarrow F - \{f\}$; $F_{pool} \leftarrow F_{pool} \cup \{f\}$
11: **end for**
12: **end if**
13: **end for**
14: $F \leftarrow F_{pool}$
15: **end while**

表 1: 検定に用いる表の例
Table 1: Example of table used for the statistical test

フィルタ \ 集計数	True	False
フィルタ 1	5	3
フィルタ 2	3	10
フィルタ 3	8	1
フィルタ 4	10	1

タスクの結果の集計結果を表 1 のようにまとめ，それを元に検定を行う．例えば，表 1 のフィルタ 2 とフィルタ 4 について，フィッシャーの正確確率検定を用いて検定すると，p = 0.001349 となり，0.05 未満なので有意差があることがわかる．

4.3 フィルタの選別の手順

本手法の詳細なアルゴリズムを Algorithm 2 に示す．Algorithm 2 は Algorithm 1 に比べ，他の結果に依存するフィルタの対応や検定に用いる表（Algorithm 2 の table）を考慮するなど，より詳細なものになっている．

本手法では，ランダムに実行順序を決められた（2 行目）タスクの内，未処理なもの（9 行目）をワーカへ処理を依頼し，結果に格納する（10 行目）．ただし，F に含まれるフィルタのどれにも当てはまらないタスクの場合にはタスクの処理をスキップする（7-8 行目）．そして，タスク処理をスキップしなかった場合には，そのタスクの処理結果を踏まえ，検定に用いる表を更新し（17-19 行目），F に含まれるフィルタの各ペアについて検定し（20-21 行目），有意差があった場合には適合率の低い方を F から取り除き，F_{pool} に移す（22-23 行目）．そして，F に含まれるフィルタのいずれかにマッチするタスクの全てを処理した時点で，その時点の F_{pool} を F に代入する．この手順をタスクの全てが処理されるまで繰り返す（3 行目）．ただし，タスクを行った結果，あらかじめ判定できないフィルタに対して新たにマッチするタスクが判明した場合や，動的にタスク数が増加した場合など状況が変わった場合（12 行目）には，新しく追加されたタスクを既に生成されているランダムな順序列のランダムな位置に挿入し（13 行目），タスクの実行順序と処理結果のみを引き継ぎ，はじめから行う（14-15 行目）．

5. シミュレーション

本研究では，大きく分けて 2 種類のシミュレーションを行った．本節では，まず，提案手法の有効性を確かめるためのシミュレーションについて説明する．特に，3.4 節で説明した一般化選択フィルタの応用 (1) のよりよい中間結果の入手について，提案手法が有効であることを確かめる．次に，提案手法を用いた場合とそう

Algorithm 2 提案手法の詳細なアルゴリズム
Detailed algorithm of the proposed method

Require: T, F_{ALL}
Ensure: results[0, ... , $|T|-1$]
1: $F \leftarrow F_{ALL} \cup \{\sigma_\phi\}$
2: tasks ← order_randomly(T)
3: **while** ∃i $(0 \leq i < |T|)$ *s.t.* results[i] = NULL **do**
4: $F_{pool} \leftarrow \phi$
5: table ← initialize_table(F)
6: **for** i in [0 ... tasks.length - 1] **do**
7: matched_f ← $\{f | f \in F,\ s.t.\ f$ matches tasks[i]$\}$
8: **if** matched_f != ϕ **then**
9: **if** results[i] = NULL **then**
10: (results[i], situation_change) ← ask(tasks[i])
11: **end if**
12: **if** situation_change **then**
13: add_tasks_random_order(tasks, new_tasks)
14: $F \leftarrow F_{ALL} \cup \{\sigma_\phi\}$
15: break
16: **end if**
17: **for all** f ∈ matched_f **do**
18: table[f.id][results[i]] + +
19: **end for**
20: **for all** $f1, f2$ *s.t.* $f1 \in F \wedge f2 \in F \wedge f1 \neq f2$ **do**
21: **if** exist_significant_diff(table, $f1$, $f2$) **then**
22: $f \leftarrow$lower_precision($f1$, $f2$)
23: $F \leftarrow F - \{f\}$; $F_{pool} \leftarrow F_{pool} \cup \{f\}$
24: **end if**
25: **end for**
26: **end if**
27: **end for**
28: **if** !situation_change **then**
29: $F \leftarrow F_{pool}$
30: **end if**
31: **end while**

でない場合の処理時間のシミュレーションについて説明する．

有効性を確かめるためのシミュレーションは，他の結果に依存するフィルタを含まない場合と含む場合の 2 種類を行った．含まない場合のシミュレーションでは，メールのスパム判定を行う問題を，含む場合のシミュレーションでは，実際に行われたクラウドソーシングの例として，被災地における建物の倒壊判定を行う問題を扱った．

本節では，各問題について，まず，シミュレーションの設定として用いるデータセットとフィルタについて説明し，その後，提案手法を適用した場合とそうでない場合の再現率の推移と，フィルタの適用状況の推移を示し，それらについて説明する．

5.1 問題 1. メールのスパム判定

5.1.1 シミュレーション設定

CS MINING GROUP[11] にある，多数のメールとそれらがスパムであるかどうかを判定したデータを用いた．メールの総数は，4327 件であり，その内の約 31.8%（1378 件）がスパムと判定されている．

表 2 に問題 1 のシミュレーションに用いた一般化選択フィルタの一覧を示す．表 2 のフィルタの全ては他の結果に依存しないため，各フィルタがマッチするかどうかを全てのタスクについてあらかじめ判定できる．

フィルタの選別における検定では，両側検定で有意水準 0.05 を用いた．

5.1.2 シミュレーション結果

図 3 に提案手法を適用した場合とランダムな順序でタスクを処理した場合の再現率の推移を示す．図 3 から提案手法を用いたほうが早い時点で再現率が上昇していることがわかり，提案手法がよりよい中間結果の入手という観点で有効であると考えられる．

図 4 に各フィルタの適用の推移を示す．図 4 は各タスクを処理した時点で各フィルタが F に含まれていたかどうかの推移を表したものであり，値（線）なしは当該フィルタにマッチするタスク

表 2: 問題 1 のシミュレーションに用いるフィルタ
Table 2: Filters used for the simulation of the problem 1

	ヒューリスティクス	適合率	再現率
f0	（空フィルタ）	31.8%	100%
f1	本文に「http」が含まれているならばスパム	30.6%	83%
f2	本文に「$」が含まれているならばスパム	49.8%	24.8%
f3	本文に「click」が含まれているならばスパム	66.1%	54%
f4	件名に「spam」が含まれているならばスパム	74.7%	27.6%

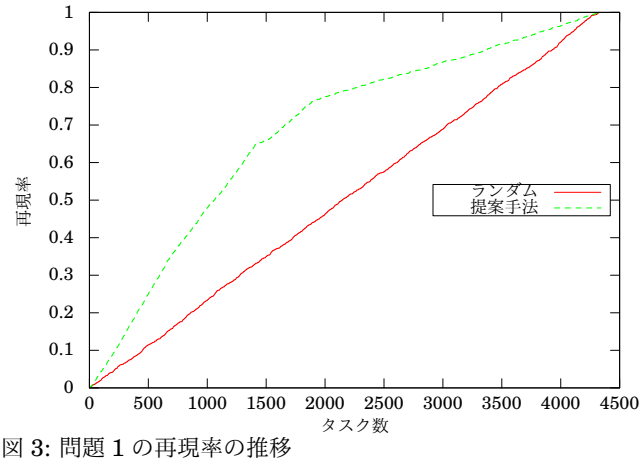

図 3: 問題 1 の再現率の推移
Figure 3: Transition of recalls in the problem 1

が無くなった状態を表している．図 4 からフィルタの選別が適合率の高い順に行われていることが確認でき，本手法におけるフィルタの選別が意図した通り行われていることがわかる．

また，図 4 から，f0 と f1 に有意差がみられずに最後までタスクが処理されたことが確認できる．f1 のように F に含まれるフィルタは必ずしも優れているとは限らないが，そのような場合でも本手法では結果に悪影響がでないことが図 3 と図 4 からわかる．

5.2 問題 2. 建物の倒壊判定

5.2.1 シミュレーション設定

2013 年に発生した台風 26 号の伊豆大島における台風被害の実データを用いて，建物の倒壊判定（図 5）をクラウドソーシングする際に提案手法を適用するシミュレーションを行う．シミュレーションの際の倒壊の判定結果は人手で付与したものを用いる．与えられた建物座標は 11350 件であり，その内の約 1%（118 件）が倒壊と判定された．

表 3 にシミュレーションに用いた一般化選択フィルタの一覧を示す．各フィルタで用いられるヒューリスティクスの実装の詳細は本論文の議論に関連しないため省略する．f3 は，他のタスクにおける建物の倒壊判定に応じて，マッチするかどうかが変わるので，他の結果に依存するフィルタである．

フィルタの選別における検定では，両側検定で有意水準 0.05 を用いた．

5.2.2 シミュレーション結果

図 6 に提案手法を適用した場合とランダムな順序でタスクを処理した場合それぞれの再現率の推移を示す．図 6 から提案手法を用いたほうが早い時点で再現率が上昇していることがわかり，他の結果に依存するフィルタを含む場合であっても，提案手法が有効であることが確認できる．

図 6 から提案手法における再現率は，2600 タスク程度を処理した時点で 90%を超えており，最も優れたフィルタの再現率の 85.6%よりも高いことがわかる．これは，この時点で最も優れたフィルタ（f3）と二つ目に優れたフィルタ（f2）によって抽出されるタスクの多くの処理を終えており，最も優れたフィルタと二つ目に優れたフィルタが抽出するタスク集合の和集合の再現率（91%程

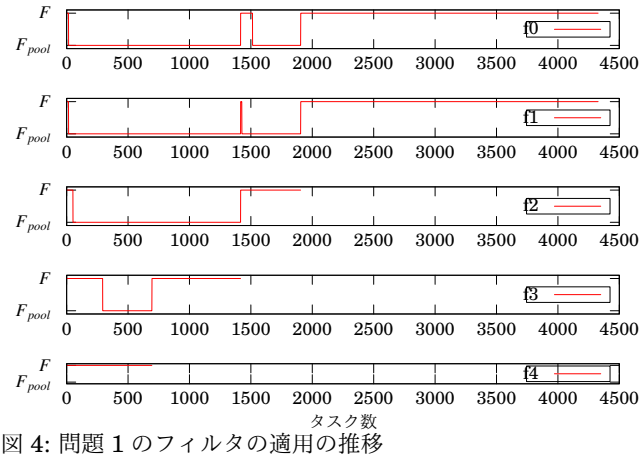

図 4: 問題 1 のフィルタの適用の推移
Figure 4: Transition of filter application status in the problem 1

図 5: 建物の倒壊判定タスク例
Figure 5: Sample task for finding a destroyed building

度）に近づいているからだと考えられる．

図 7 に各フィルタの適用の推移を示す．図 7 から 1500 タスク程度から 2400 タスク程度の区間では，f2 と f3 の適用が交互に行われていることがわかる．処理した時点で f2 もしくは f3 が F に含まれている状況が続く．これは，f2 によって抽出されたタスクによって壊れている建物が発見されることで，f3 にマッチするタスクが増え始め，f3 が優れていると判断されるからである．そして，その時点で判明している f3 にマッチするタスクを終えると，f2 にマッチするタスクを行い，それによって f3 にマッチするタスクを見つけるとそれらを行うという繰り返しになっている．

また，タスクの結果によって f3 にマッチするタスクが見つかると，瞬間的に f3 が再び F に含まれることが図 7 からわかり，他の結果に依存する場合であっても，適合率の優れたフィルタができる限り優先的に適用されていることが確認できる．

提案手法の再現率が顕著に上昇し始めるのが 1500 タスク程度にまで遅れるのは，f2 と f3 が有意に優れていることがわかるのにそれだけのタスク数を要したからである．しかし，他の結果に依存するフィルタである f3 の選別が遅れるのはともかく，f2 の選別が遅いのは望ましくない．他のフィルタに比べ優れているフィルタがあらかじめ絞れている場合や，今回のシミュレーションのように F_{ALL} の要素数が少ない場合には，全体からランダムにタスクを選ぶのではなく，有意差が早く見つかるようにサンプルが足りないフィルタについてマッチするタスク中からランダムに選んで，先に処理することで，効率よく選別が可能であると考えられる．

5.3 処理時間についてのシミュレーション

3.5 節で述べたように，提案手法を用いると全体の計算量は増加してしまうが，同じ再現率時点での処理時間を比較した場合には提案手法のほうが短くなることが予想される．本節では，どのような条件下であれば提案手法を用いたほうがそうでない場合に比

表 3: 問題 2 のシミュレーションに用いるフィルタ
Table 3: Filters used for the simulation of the problem 2

	ヒューリスティクス	適合率	再現率
f0	（空フィルタ）	1.0%	100%
f1	建物周辺の画素の色がまばらならば壊れている	0.9%	44%
f2	建物周辺の画素の多くが茶色ならば壊れている	4.8%	53.4%
f3	壊れている建物の 100m 以内ならば壊れている	28%	85.6%

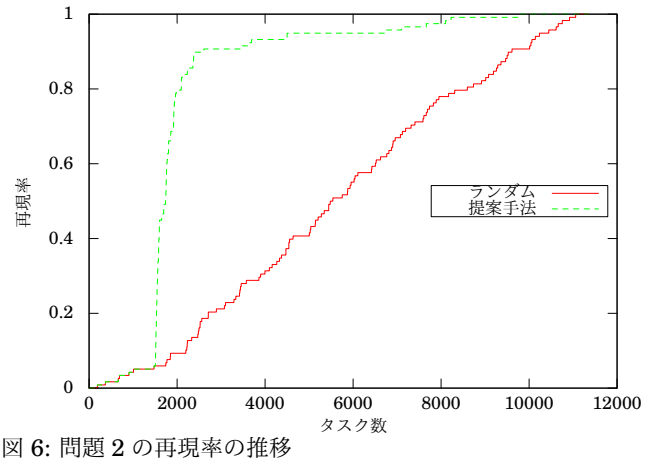

図 6: 問題 2 の再現率の推移
Figure 6: Transition of recalls in the problem 2

べ，処理時間が短くなるかについてシミュレーションを行った結果について説明する．

本シミュレーションでは処理時間の計算を簡単化するために，タスクは同時に処理されず，誰もタスクを行わない待ち時間は存在しないこととする．

まず，提案手法を用いずにタスクを処理した場合と用いた場合で，それぞれ再現率 r になるまでに必要なタスク数を k_1, k_2 とする．また，一つのタスクを処理するのにかかる平均時間を t 秒，一つのタスクについて一つのフィルタの条件を判定するのにかかる平均時間を t_f 秒とする．そして，F_{ALL} 中のフィルタの数を n_f とすると，提案手法が満たすべき条件式として次の式が得られる．

$$t \cdot k_2 + t_f \cdot n_f \cdot k_2 < t \cdot k_1 \tag{1}$$

ここで，n_f が何件であれば，条件を満たすのかについて注目し，式を整理すると次の式が得られる．

$$n_f < \frac{t}{t_f}\left(\frac{k_1}{k_2} - 1\right) \tag{2}$$

また，データセットの総タスク数を n_t，p に関する選択率を s とし．フィルタを用いた場合の再現率 r 時点における平均適合率 $(s \cdot n_t \cdot r/k_2)$ を p_{avg} とする．そして，$r = k_1/n_t$ であることを考慮すると，次の式が得られる．

$$n_f < \frac{t}{t_f}\left(\frac{p_{avg}}{s} - 1\right) \tag{3}$$

(3) 式から，提案手法のほうが処理時間が短くなるためには，フィルタの平均適合率が選択率より高いことが必要条件であることがわかる．この必要条件を満たすことは容易であると考えられる．なぜなら，提案手法では空フィルタよりも有意に適合率の低いフィルタは除去されることを考慮すると，平均適合率は選択率以上になることが予想されるからである．

図 8 は (3) 式を元に，$t = 1, t_f = 0.001$ の場合の選択率によって推移する適用可能なフィルタ数を示したものである．図 8 から選択率の低いデータセットほど提案手法がより効果的であることがわかる．

図 9 は (2) 式を元に，問題 1 と問題 2 の再現率によって推移する適用可能なフィルタ数を示したものである．

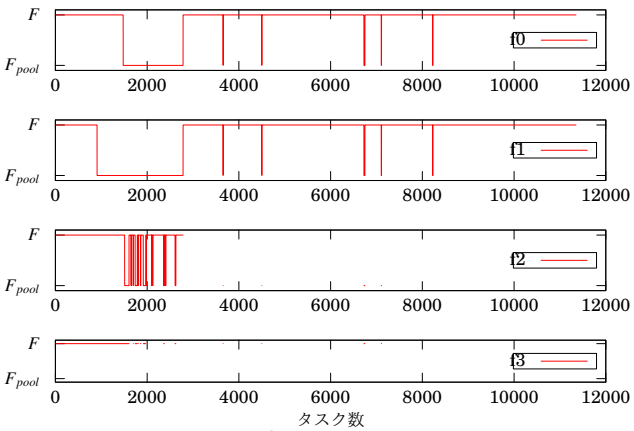

図 7: 問題 2 のフィルタの適用の推移
Figure 7: Transition of filter application status in the problem 2

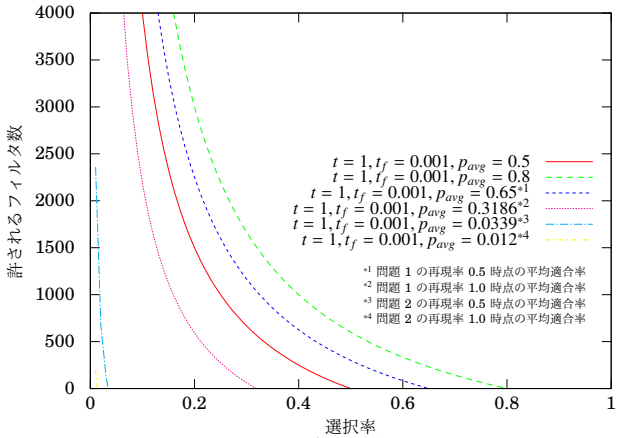

図 8: 選択率によって推移する適用可能なフィルタ数
Figure 8: Selection ratios and the number of filters that can be handled

図 9 から再現率 1.0 付近では，許されるフィルタの数が少なくなっており，再現率 1.0 を早く達成するという目的には，提案手法が向いていないと考えられる．しかし，問題 1 のシミュレーションに用いたフィルタの数 4 個（空フィルタを除く）を 99%の区間で上回っており，問題 2 のシミュレーションに用いたフィルタの数 3 個（空フィルタを除く）を 95%の区間で上回っており，提案手法の利用価値は十分にあると考えられる．

6. まとめと今後の課題

本論文は，クラウドソーシングシステムにおけるヒューリスティクスの動的な適用を行うための枠組みの提案を行った．そのために，ヒューリスティクスを一般化選択フィルタとしてモデル化し，統計的に有意に優れているヒューリスティクスの選別を行えるようにした．また，実データを用いたシミュレーションによって，提案手法が一般化選択フィルタの応用の一つである「よりよい途中結果の入手」において有効であることを確認した．

今後の課題として，提案手法の効果に関する理論的論証や，再現率の向上だけでなくタスク UI の切り替え等の一般化選択フィルタの他の応用についても本手法を利用可能であるかどうかの検証が挙げられる．

また，本手法の発展として，実行時の動的なヒューリスティクスの追加や，フィルタの組み合わせを自動生成し，候補に加えることも今後の課題として挙げられる．実行時のヒューリスティクスの追加は，リクエスタがクラウドソーシングの途中結果を見て，

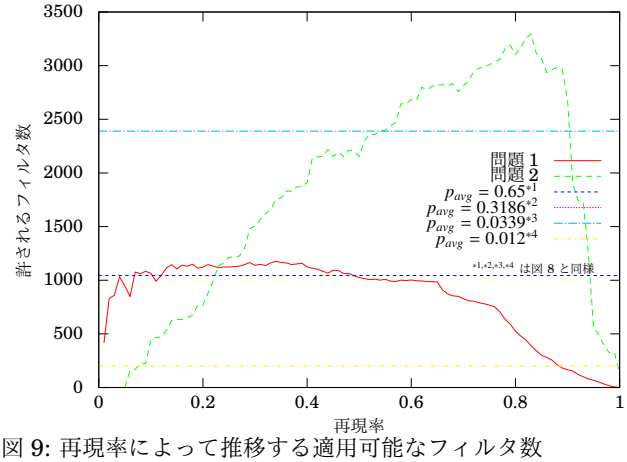

図 9: 再現率によって推移する適用可能なフィルタ数
Figure 9: Recalls and the number of filters that can be handled

新たにヒューリスティクスを追加するという場合に役立つと考えられる．フィルタの組み合わせの生成については，多数存在する組み合わせから，いかにして優れていると予測される組み合わせを見つけるかが課題である．

[謝辞]

本論文の一部は科研費基盤研究 (#25240012) の支援による．

[文献]

[1] B. H. Bloom. “Space/time trade-offs in hash coding with allowable errors”. Communications of the ACM, 13(7): 422-426, 1970.

[2] U. S. Chakravarthy, J. Grant, J. Minker. “Logic-based approach to semantic query optimization”. ACM Transactions on Database Systems (TODS) 15(2): 162-207, 1990.

[3] A. Doan, R. Ramakrishnan, A. Y. Halevy. “Crowdsourcing systems on the world-wide web”. Communications of the ACM 54(4): 86-96, 2012.

[4] G. Giacinto, F. Roli. “Methods for dynamic classifier selection”. Proceedings of International Conference on Image Analysis and Processing 1999: 659-664, 1999.

[5] A. Marcus, E. Wu, D. Karger, S. Madden, R. Miller. “Human-powered sorts and joins”. PVLDB 5(1): 13-24, 2011.

[6] Y. Shinagawa, A. Morishima, S. Nakamura, T. Terada. “Human Computation Environment Embedded in Living Spaces for Crowdsourcing Task Processing”. インタラクション 2014, 2014.

[7] K. N. Truong, T. Shihipar, D. Wigdor. “Slide to X: unlocking the potential of smartphone unlocking”. Proceedings of the 32nd annual ACM conference on Human factors in computing systems: 3635-3644, 2014

[8] R. Vaish, K. Wyngarden, J. Chen, B. Cheung, M. Bernstein. “Twitch crowdsourcing: crowd contributions in short bursts of time”. Proceedings of the 32nd annual ACM conference on Human factors in computing systems: 3645-3654, 2014

[9] M. Woźniak, M. Graña, E. Corchado. “A survey of multiple classifier systems as hybrid systems”. Information Fusion 16: 3-17, 2014.

[10] K. Woods, P. W. Kegelmeyer Jr, K. Bowyer. “Combination of multiple classifiers using local accuracy estimates”. Proceedings of Computer Vision and Pattern Recognition 1996, 1996 IEEE Computer Society Conference: 391-396, 1996.

[11] http://csmining.org/index.php/spam-email-datasets-.html

権守 健嗣 Kenji GONNOKAMI
2013 年 3 月筑波大学情報学群情報メディア創成学類卒業．2015 年 3 月筑波大学大学院システム情報工学研究科修了．2015 年 4 月より株式会社 VASILY 勤務．

森嶋 厚行 Atsuyuki MORISHIMA
1998 年筑波大学大学院工学研究科修了．現在，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教授．クラウドソーシングシステムに興味を持つ．日本データベース学会理事，情報処理学会 DBS 研究会主査，情報処理学会論文誌 TOD 共同編集委員長，The VLDB Journal Associate Editor，情報処理学会，電子情報通信学会，ACM, IEEE-CS 各会員

歳森 敦 Atsushi TOSHIMORI
1992 年筑波大学大学院社会工学研究科中退．現在，筑波大学図書館情報メディア系教授．博士（工学）．日本建築学会建築計画委員会地域施設小委員会幹事，日本建築学会，日本図書館情報学会各会員

北川 博之 Hiroyuki KITAGAWA
1978 年東京大学理学部物理学科卒業．1980 年同大学理学系研究科修士課程修了．日本電気（株）勤務の後，1988 年筑波大学電子・情報工学系講師．同助教授を経て，現在，筑波大学システム情報系教授，ならびに計算科学研究センター教授．理学博士（東京大学）．データベース，情報統合，データマイニング，情報検索等の研究に従事．日本データベース学会会長，情報処理学会フェロー，電子情報通信学会フェロー，ACM，IEEE，日本ソフトウェア科学会，各会員．